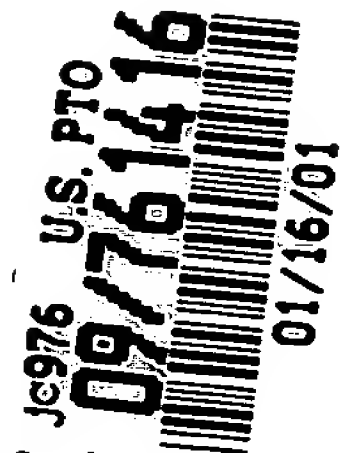


日 本 国 特 許 庁  
PATENT OFFICE  
JAPANESE GOVERNMENT



別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日  
Date of Application: 2000年 1月18日

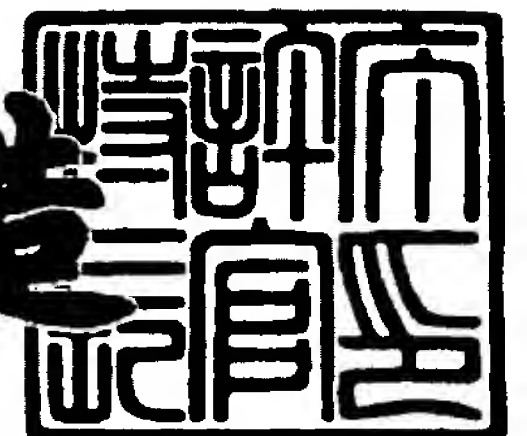
出 願 番 号  
Application Number: 特願2000-013957

出 願 人  
Applicant(s): ソニー株式会社

2000年11月17日

特許庁長官  
Commissioner,  
Patent Office

及川耕造



出証番号 出証特2000-3095276

【書類名】 特許願

【整理番号】 9900691003

【提出日】 平成12年 1月18日

【あて先】 特許庁長官殿

【国際特許分類】 H04L 12/40

【発明者】

【住所又は居所】 東京都品川区北品川 6 丁目 7 番 3 5 号 ソニー株式会社  
内

【氏名】 堀口 麻里

【特許出願人】

【識別番号】 000002185

【氏名又は名称】 ソニー株式会社

【代表者】 出井 伸之

【代理人】

【識別番号】 100067736

【弁理士】

【氏名又は名称】 小池 晃

【選任した代理人】

【識別番号】 100086335

【弁理士】

【氏名又は名称】 田村 榮一

【選任した代理人】

【識別番号】 100096677

【弁理士】

【氏名又は名称】 伊賀 誠司

【手数料の表示】

【予納台帳番号】 019530

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9707387

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 情報処理装置及び方法、媒体

【特許請求の範囲】

【請求項 1】 ネットワークを介して他の情報処理装置と接続可能であるとともに、所定の機能を実行する 1 以上の機能実行手段と、上記機能実行手段の使用予定に関する第 1 の情報を格納する格納手段とを少なくとも有する情報処理装置において、

上記機能実行手段の使用予定に関する第 1 の情報に含まれない第 2 の情報入手する入手手段と、

上記第 2 の情報を所定のブロック形式とし、上記第 1 の情報の付加情報として上記格納手段に格納させる制御手段とを有することを特徴とする情報処理装置。

【請求項 2】 上記第 1 の情報は、所定のデジタルインターフェイスフォーマットにて規定された上記使用予定の情報であり、

上記第 2 の情報は、上記使用予定の詳細を表す表示可能なテキスト情報であることを特徴とする請求項 1 記載の情報処理装置。

【請求項 3】 上記第 2 の情報は、ローテキストと当該ローテキストのコーディングを表すキャラクタコード及び言語コードを含むことを特徴とする請求項 2 記載の情報処理装置。

【請求項 4】 上記制御手段は、上記ネットワークに接続された他の情報処理装置の上記格納手段に対して、上記第 1 の情報及び第 2 の情報の書き込みと読み出しを制御する機能を有することを特徴とする請求項 1 記載の情報処理装置。

【請求項 5】 上記制御手段は、上記ネットワークに接続された他の情報処理装置の制御手段からの指令に基づいて、上記格納手段に対する上記第 1 の情報及び上記第 2 の情報の書き込みと読み出しを制御する機能を有することを特徴とする請求項 1 記載の情報処理装置。

【請求項 6】 上記入手手段は、装置の使用者から入力された上記機能実行手段の使用予定に関する設定情報から上記第 2 の情報を生成する機能を有することを特徴とする請求項 1 記載の情報処理装置。

【請求項 7】 上記入手手段は、上記ネットワークに接続された他の情報処理装置の格納手段に格納されている上記第 2 の情報を入力する機能を有することを特徴とする請求項 1 記載の情報処理装置。

【請求項 8】 ネットワークを介して他の情報処理装置と接続可能であるとともに、所定の機能を実行する 1 以上の機能実行手段と、上記機能実行手段の使用予定に関する第 1 の情報を格納する格納手段とを少なくとも有する情報処理装置の情報処理方法において、

上記機能実行手段の使用予定に関する第 1 の情報に含まれない第 2 の情報を入力し、

上記第 2 の情報を所定のブロック形式とし、上記第 1 の情報の付加情報として上記格納手段に格納させる制御を行う

ことを特徴とする情報処理方法。

【請求項 9】 上記第 1 の情報は、所定のデジタルインターフェイスフォーマットにて規定された上記使用予定の情報であり、

上記第 2 の情報は、上記使用予定の詳細を表す表示可能なテキスト情報であることを特徴とする請求項 8 記載の情報処理方法。

【請求項 10】 上記第 2 の情報は、ローテキストと当該ローテキストのコーディングを表すキャラクタコード及び言語コードを含むことを特徴とする請求項 9 記載の情報処理方法。

【請求項 11】 上記ネットワークに接続された他の情報処理装置の上記格納手段について、上記第 1 の情報及び第 2 の情報の書き込みと読み出しを制御することを特徴とする請求項 8 記載の情報処理方法。

【請求項 12】 上記ネットワークに接続された他の情報処理装置からの指令に基づいて、上記格納手段に対する上記第 1 の情報及び上記第 2 の情報の書き込みと読み出しを制御することを特徴とする請求項 8 記載の情報処理方法。

【請求項 13】 装置の使用者から入力された上記機能実行手段の使用予定に関する設定情報から上記第 2 の情報を生成することを特徴とする請求項 8 記載の情報処理方法。

【請求項 1 4】 上記ネットワークに接続された他の情報処理装置の格納手段に格納されている上記第 2 の情報を入手することを特徴とする請求項 8 記載の情報処理方法。

【請求項 1 5】 1 以上の所定の機能の使用予定に関する第 1 の情報に含まれない第 2 の情報を入手するステップと、

上記第 2 の情報を所定のブロック形式とし、上記第 1 の情報の付加情報として格納手段に格納させるステップとを含むことを特徴とするプログラムを情報処理装置に実行させる媒体。

【発明の詳細な説明】

【 0 0 0 1 】

【発明の属する技術分野】

本発明は、例えば I E E E 1 3 9 4 シリアルデータバスを介して他の情報処理装置と接続される情報処理装置において、内蔵するサブユニットを重複することなく確実に制御することができるようにする情報処理装置及び方法、媒体に関する。

【 0 0 0 2 】

【従来の技術】

近年は、例えば I E E E (The Institute of Electrical and Electronics Engineers) で規格化された I E E E 1 3 9 4 シリアルデータバスを用いるネットワークを介して、相互に情報を伝達できるようにした A V 機器が開発されている。このネットワークシステムにおいては、所定のデジタルインターフェイスコマンド (A V / C Command Transaction Set: 以下 A V / C コマンドと略称する) を用いて、上述のネットワークに接続されている A V 機器を相互に制御することが可能である。

【 0 0 0 3 】

図 3 4 には、I E E E 1 3 9 4 シリアルデータバスを用いたネットワークの一例を示す。この図 3 4 のネットワークシステムにおいては、I E E E 1 3 9 4 シリアルデータバス 8 0 (以下、単にバス 8 0 とする) を用いることにより、例えばデジタル衛星放送を受信する I R D (Integrated Receiver Decoder) 7 1

で受信された映像信号を、バス 8 0 介して接続されている D V C R (Digital Video Cassette Recorder) 8 1 で録画することができる。さらにこれらの I R D 7 1、D V C R 8 1 を用いて、いわゆる予約録画をすることも可能である。

## 【 0 0 0 4 】

この装置で予約録画を行う場合には、例えば I R D 7 1 内に設けられるコントローラ 7 2 によって I R D 7 1 自身と D V C R 8 1 とが制御される。すなわち、予約録画の設定（チャンネル、開始時刻等の設定）は、I R D 7 1 に対して行われる。そして、設定された開始時刻になると、コントローラ 7 2 は、I R D 7 1 内のデジタルチューナサブユニット 7 3 に対して内部的に制御し、上記予約されている（設定されている）チャンネルを選局させ、C S アンテナ 7 4 で捉えた信号の中から当該デジタルチューナサブユニット 7 3 で受信した映像信号等をバス 8 0 を介して D V C R 8 1 に出力させる。また同時に、コントローラ 7 2 からは、バス 8 0 を介して D V C R 8 1 内に設けられる V C R サブユニット 8 4 に対して録画開始のコマンドが送信される。このときの V C R サブユニット 8 4 は、コントローラ 7 2 から送信されてきた録画開始コマンドに対応して、デジタルチューナサブユニット 7 3 から送信されてきた映像信号を図示しない磁気テープ等の記録媒体へ記録する。以上により、図 3 4 のネットワークにおける予約録画が行われる。

## 【 0 0 0 5 】

また、D V C R 8 1 もコントローラ 8 2 を備えており、当該コントローラ 8 2 は、例えば内蔵のアナログチューナサブユニット 8 3 で受信された映像信号を V C R サブユニット 8 4 で記録するなどの制御を行う。

## 【 0 0 0 6 】

なお、バス 8 0 に接続されている I R D 7 1 および D V C R 8 1 のような電子機器はユニットと呼ばれており、ユニット間においては、A V / C コマンド (A V / C Command Transaction Set) の一般的な仕様 (A V / C Digital Interface Command Set General Specification、以下、A V / C ジェネラルと略称する) で規定されている記述 (Descriptor、以下、適宜ディスクリプタと呼ぶ) を用いて、各ユニットに記憶されている情報を相互に読み書きすることが可能となされている



。A V / C ジェネラルの詳細については、「<http://www.1394ta.org/>」に公開されている。また、ユニットが有する機能は、サブユニットと呼ばれており、図 3 4 の例では、I R D 7 1 のデジタルチューナサブユニット 7 3、D V C R 8 1 の V C R サブユニット 8 4 等がサブユニットとなる。

#### 【 0 0 0 7 】

##### 【発明が解決しようとする課題】

ところで上述したように、D V C R 8 1 の動作を、バス 8 0 を介して接続されている他の機器（図 3 4 の例の場合、I R D 7 1）が制御できるようなネットワークシステムの場合は、いわゆるダブルブッキングが生ずる虞がある。

#### 【 0 0 0 8 】

例えば、デジタル衛星放送の録画予約（録画予約 A とする）を I R D 7 1 に入力すると、その予約情報は、I R D 7 1 のコントローラ 7 2 に記憶されることになる。但し、この時点（録画予約の時間になる前の時点）では、I R D 7 1 から D V C R 8 1 に対してその録画予約に関連した A V / C コマンドが送信されていないため、当該 D V C R 8 1 は、I R D 7 1 に録画予約の設定がなされていることを知らない。その後、例えば録画予約 A の録画時刻に重複する時刻において放送される地上波アナログ放送の録画予約（録画予約 B とする）が D V C R 8 1 に設定された場合、当該 D V C R 8 1 のコントローラ 8 2 は、I R D 7 1 に設定されている録画予約 A に関する情報を得ていないので、時刻が重複しているか否かを知ることができず、上記録画予約 B を受け付けて記憶してしまう。

#### 【 0 0 0 9 】

したがって、録画予約 A および録画予約 B で指定された時刻になると、D V C R 8 1 の V C R サブユニット 8 4 には、I R D 7 1 のデジタルチューナサブユニット 7 3 及び D V C R 8 1 のアナログチューナサブユニット 8 3 の両方から映像信号が供給されると共に、I R D 7 1 のコントローラ 7 2 及び D V C R 8 1 のコントローラ 8 2 の両方による制御がなされてしまうことになり、不都合が生じてしまう。

#### 【 0 0 1 0 】

上述した従来のネットワークシステムにおいて、このようなダブルブッキング



による不都合が発生するのは、例えば、バス 8 0 を介して接続されている各 A V 機器（ユニット）が他の A V 機器による予約情報等の詳細を入手できないこと、或いは、各 A V 機器が他の A V 機器に対して自己が持つ予約情報等の詳細を知らせる手段がないこと、また、ユーザが新たな予約の入力を行う場合に既に設定されている予約情報の詳細を知り得ないこと、更には、各 A V 機器やユーザが予約情報の詳細を入力したくても入力できないことなどに起因している。

## 【 0 0 1 1 】

そこで、本発明はこのような状況に鑑みてなされたものであり、ユーザや各機器が予約情報等の詳細を入力できること、各機器やユーザが他の機器による予約情報等の詳細を入手できる（知り得る）こと、及び、各機器が他の機器やユーザに対して自己が持つ予約情報等の詳細を知らせる手段を実現することにより、ダブルブッキングの発生を抑止可能となるだけでなく、分かり易く且つ使い易いネットワークシステムを構築可能とする、情報処理装置及び方法、媒体を提供することを目的とする。

## 【 0 0 1 2 】

## 【課題を解決するための手段】

本発明の情報処理装置は、上述した課題を解決するために、ネットワークを介して他の情報処理装置と接続可能であるとともに、所定の機能を実行する 1 以上の機能実行手段と、上記機能実行手段の使用予定に関する第 1 の情報を格納する格納手段とを少なくとも有する情報処理装置において、上記機能実行手段の使用予定に関する第 1 の情報に含まれない第 2 の情報を入手する入手手段と、上記第 2 の情報を所定のブロック形式とし、上記第 1 の情報の付加情報として上記格納手段に格納させる制御手段とを有することを特徴とする。

## 【 0 0 1 3 】

本発明の情報処理方法は、上述した課題を解決するために、ネットワークを介して他の情報処理装置と接続可能であるとともに、所定の機能を実行する 1 以上の機能実行手段と、上記機能実行手段の使用予定に関する第 1 の情報を格納する格納手段とを少なくとも有する情報処理装置の情報処理方法において、上記機能実行手段の使用予定に関する第 1 の情報に含まれない第 2 の情報を入手し、上記

第 2 の情報を所定のブロック形式とし、上記第 1 の情報の付加情報として上記格納手段に格納させる制御を行うことを特徴とする。

【 0 0 1 4 】

本発明の媒体は、上述した課題を解決するために、1 以上の所定の機能の使用予定に関する第 1 の情報に含まれない第 2 の情報を入手するステップと、上記第 2 の情報を所定のブロック形式とし、上記第 1 の情報の付加情報として格納手段に格納させるステップとを含むことを特徴とするプログラムを情報処理装置に実行させることを特徴とする。

【 0 0 1 5 】

【発明の実施の形態】

本発明の好ましい実施の形態について、図面を参照しながら説明する。なお、本明細書において、システムの用語は、複数の装置、手段などにより構成される全体的な装置を意味するものである。

【 0 0 1 6 】

図 1 には、本発明実施の形態のネットワークシステムの概略構成を示す。

【 0 0 1 7 】

この図 1 に示すネットワークシステムは、IEEE 1394 シリアルデータバス 2（以下、バス 2 とする）を介して接続されている IRD 1 及び DVCR（例えば D-VHS 等のレコーダ）3 から構成されている。もちろん、このバス 2 には IRD 1 や DVCR 3 以外に、例えば、パーソナルコンピュータ、ハードディスクドライブ、CD プレーヤ、モニタ、デジタルビデオカメラ、MD（商標）プレーヤ等の IEEE 1394 端子を備える電子機器を接続することが可能である。

【 0 0 1 8 】

IRD 1 のコントローラ 11 は、ユーザからの選局操作や録画予約操作等を受け付けて、IRD 1 の全体を制御する。また、当該コントローラ 11 は、所定のデジタルインターフェイスコマンドとして、前記 AV/C コマンドを用いて DVCR 3 を制御する。CS アンテナ 13 は、図示せぬ通信衛星を介して送信されてくるデジタル衛星放送のデジタル信号を受信して、チューナサブユニット

1 2 に出力する。チューナサブユニット 1 2 は、コントローラ 1 1 の制御に基づいて、C S アンテナ 1 3 から入力されたデジタル信号から所望のチャンネルの信号を抽出し、バス 2 を介して D V C R 3 の V C R サブユニット 3 3 に出力する。さらに、コントローラ 1 1 は、D V C R 3 の B B S (Bulletin Board Subunit) 3 4 に記録されている情報を検索可能となっている。

## 【 0 0 1 9 】

I R D 1 のサブユニットの一つである B B S 1 4 は、コントローラ 1 1 が受け付けて、確定した録画予約等の情報（詳細は図 2 により後述する）を格納する。

## 【 0 0 2 0 】

D V C R 3 のコントローラ 3 1 は、ユーザからの再生指示の操作や録画予約操作等を受け付けて、D V C R 3 の全体を制御する。また、コントローラ 3 1 は、B B S 3 4 に記録されている情報を検索可能である。アナログチューナサブユニット 3 2 は、コントローラ 3 1 の制御に基づいて、入力されるアナログ信号から所定のチャンネルの信号を抽出し、V C R サブユニット 3 3 に出力する。

## 【 0 0 2 1 】

V C R サブユニット 3 3 は、アナログチューナサブユニット 3 2 から入力された映像信号、又は、バス 2 を介して入力される I R D 1 のチューナサブユニット 1 2 からの映像信号を図示せぬ磁気テープに記録する。

## 【 0 0 2 2 】

B B S 3 4 は、D V C R 3 に関わる録画予約等の情報（詳細は図 2 により後述する）を格納する。

## 【 0 0 2 3 】

この図 1 に示したネットワークシステムにおいて、デジタル衛星放送の録画予約を行う場合、ユーザは、I R D 1 に対して録画予約の設定（チャンネル及び録画開始時刻等の設定）を入力する。そして、その録画予約の時刻やチャンネル等が既に予約されているものでない場合、当該入力された録画予約が認められて、その情報が I R D 1 の B B S 1 4 に書き込まれることになる。

## 【 0 0 2 4 】

ここで、I R D 1 に対してユーザにより録画予約の入力がなされた場合、当該

I R D 1 のコントローラ 1 1 は、図 2 に示すように、それぞれの録画予約についての内容を表す設定情報としてスケジュールアクション (Scheduled Action 1 ~ n) 5 1 を内部のレジスタに保持する。当該スケジュールアクション (Scheduled Action) は、スケジュールデータ (Scheduled Data) 5 2 とプログラム (Program) 5 3 からなる。また、スケジュールデータ (Scheduled Data) 5 2 は、録画の開始時間情報 (Start time)、実行時間情報 (Duration)、繰り返し情報 (Repeat information)、使用するサブユニットの情報 (Resources used) からなり、プログラム (Program) 5 3 は、サブユニットを実際に制御するためのコマンド (Command 1 ~ n) からなる。上記スケジュールデータ (Scheduled Data) 5 2 の内容 (Resource Schedule Object) は、ユーザから入力された録画予約が確定した時に、I R D 1 のコントローラ 1 1 から D V C R 3 へ送られ、当該 D V C R 3 の B B S 3 4 に書き込まれる。一方、上記プログラム (Program) 5 3 の各コマンドは、実際に録画予約の開始時間になり且つ実行時間になった時に、I R D 1 のコントローラ 1 1 から D V C R 3 のサブユニット (録画を行う際に実際に使用される各サブユニット A, B, . . . , n) に送られることになる。

## 【 0 0 2 5 】

D V C R 3 の B B S 3 4 は、図 2 に示すように、R S B (Resource Schedule Board) 6 1、及びその他のボード 6 2 から構成される。R S B 6 1 は、B B S 3 4 内において録画予約の情報のように、リソースの使用予定を書き込むためのリストである。なお、当該 R S B 6 1 で読み書きされるエントリは使用予定であって、予約された制御を実行するものではない。R S B 6 1 には、上記 I R D 1 のコントローラ 1 1 より送られてきた上記スケジュールデータ (Scheduled Data) 5 2 のオブジェクト (Start time、Duration、Repeat info、Resources) や、他のユニット (D V C R 3 のコントローラ 3 1 も含む) より送られてきた各スケジュールデータのオブジェクトが、リソーススケジュールエントリ (Resource Schedule Entry 1 ~ n) 6 3 として書き込まれる。

## 【 0 0 2 6 】

なお、D V C R 3 の B B S 3 4 に書き込まれた情報は、I R D 1 のコントローラ 1 1 だけでなく、他のユニットのコントローラ (例えば、D V C R 3 のコント

ローラ 3 1 も含む) からの要求に対応しても公開される。

【 0 0 2 7 】

図 3 には、B B S の R S B のフォーマット構成を示す。

【 0 0 2 8 】

この図 3 において、descriptor\_lengthは R S B の長さを表す。list\_typeには、R S B が読み出し専用 (Read Only) であるのか又は書き込み可能 (Write Enabled) であるのかが記述される。size\_of\_list\_specific\_informationは、list\_specific\_informationの長さを表し、list\_specific\_informationは、list\_typeによって異なるものとなる。

【 0 0 2 9 】

Write Enabled list\_specific\_infomationには、図 4 に示す情報が記述される。Write Enabled list\_specific\_infomationのnon\_info\_block\_fields\_lengthは、non info block fieldsのバイト数を表す。R S B の場合、board\_typeは図 5 に示すように” 0 1<sub>16</sub>” とされる。object\_list\_maximum\_sizeは、object list の最大の大きさを表す。object\_entry\_maximum\_numberは、listにおけるobject entriesの最大の数を表す。object\_entry\_maximum\_sizeは、object entryの最大の大きさを表す。以上のobject\_list\_maximum\_size, object\_entries\_maximum\_number, object\_entry\_maximum\_sizeは、それぞれ制限が設けられていない場合には、” 0 0 0 0<sub>16</sub>” とされる。これらの 3 つのフィールドは、コントローラがobject listまたは、object entryの容量を知る上において、有意義である。board\_type\_dependent\_information\_lengthは、board\_type\_dependent\_informationの長さを表し、board\_type\_dependent\_informationは、board typeに固有の情報を表す。

【 0 0 3 0 】

objec\_entryは図 6 に示すように記述される。objec\_entryのdescriptor\_lengthは、このdescriptorの長さを表す。entry\_typeはBoard entry Descriptorの場合、Boardを表す値” 8 0<sub>16</sub>” とされる。object\_IDは、posting\_device\_GUIDとrecord\_IDとにより構成される。posting\_deviceは、B B S に対して情報を記入 (post) したコントローラを意味し、従ってposting\_device\_GUIDは、そのG U I

Dを表す。record\_IDは、ユニット内においてイベント (Event) に対して割当てられたIDを表す。size\_of\_entry\_specific\_informationは、entry\_specific\_informationとしてのResource Schedule Entryの大きさを表す。

## 【 0 0 3 1 】

Resource Schedule Entryには、図 7 に示す情報が記述される。

## 【 0 0 3 2 】

この図 7 において、non\_info\_block\_lengthは、repeat\_informationまでのnon info block fieldsのバイト数を表す。start\_timeは、図 8 に示すように、イベント (event、この例では予約録画) が開始する年月日時分秒を表す。年 (year) は 1 6 ビットで表され、西暦を表す 4 つの数字がそれぞれ 4 ビットのBCD (Binary Coded Decimal) で表される。月 (month) は 8 ビットで表され、1 (0 1) 月から 1 2 月までの 2 桁の数字を表す 2 つの数字がそれぞれ 4 ビットのBCDで表される。日 (day) は 8 ビットで表され、1 (0 1) 日から 3 1 日までの 2 桁の数字 (日) を表す 2 つの数字がそれぞれ 4 ビットのBCDで表される。時間 (hour) は 8 ビットで表され、0 (0 0 時) から 2 4 時までの 2 桁の数字 (時刻) を表す 2 つの数字がそれぞれ 4 ビットのBCDで表される。分 (minute) は 8 ビットで表され、0 (0 0) 分から 6 0 分までの 2 桁の数字 (分) を表す 2 つの数字がそれぞれ 4 ビットのBCDで表される。秒 (second) は 8 ビットで表され、0 (0 0) 秒から 6 0 秒までの 2 桁の数字 (秒) を表す 2 つの数字がそれぞれ 4 ビットのBCDで表される。このように、start\_timeがBCDで表されることにより、その識別が容易となっている。またこの時刻は、ローカルタイムで表される。

## 【 0 0 3 3 】

イベント (event) の長さを表すDurationは、図 9 に示すように、時分秒で表される。時間 (hours) は 3 桁の時間を表す 3 つの数字がそれぞれ 4 ビットのBCDで表され、合計 1 2 ビットとなる。分 (minutes) は 2 桁の分を表す 2 つの数字がそれぞれ 4 ビットのBCDとされ、合計 8 ビットとなる。秒 (seconds) は 2 桁の秒を表す 2 つの数字がそれぞれ 4 ビットのBCDで表され、合計 8 ビットとされる。start\_timeにDurationを加算することで、イベントの終了時刻が表



される。このように、終了時刻を直接的に表現せずに、Durationとしてイベントの長さを表すようにすることで、イベントのstart\_timeが変更されたような場合においても、その終了時刻を変更するような操作が不要となり、変更処理が容易となる。

## 【 0 0 3 4 】

repeat\_information\_lengthは、repeat\_informationの長さを表している。repeat\_informationは、いつどのようにしてスケジュール (schedule) が繰り返される (この場合は、予約録画が繰り返される) かを表す。Scheduled Actionが繰り返されない場合、repeat\_information\_lengthは、" 0 0<sub>16</sub>" とされる。repeat\_informationは、選択されたrepeat\_typeによって内容が異なる。

## 【 0 0 3 5 】

repeat\_typeには、図 1 0 に示すように、Weekly scheduleの" 0 0<sub>16</sub>" とInterval scheduleの" 1 0<sub>16</sub>" とがある。スケジュール (schedule) が、週毎に繰り返される場合、ポスティングデバイス (Posting Device、この例ではIRD 1のコントローラ 1 1) は、その曜日及び繰り返されるべきイベントの数を図 1 1 に示すように表示する。このrepeat\_typeには、図 1 0 に示す値、" 0 0<sub>16</sub>" が記述される。number\_of\_eventsには、イベントの数が記述される。日曜日から土曜日までのWeekly flagsは、繰り返されるイベントが開始する週の日を表す。例えば、1 3 時 0 0 分から 3 時間のイベント (この例では予約録画) が、毎週月曜日と水曜日に開始される場合、月曜日と水曜日のフラグに" 1 " が設定され、その他のフラグには" 0 " が設定される。このように、repeat\_typeには、週毎に繰り返されるイベントが記録できるので、例えば月曜日と水曜日の絶対的な日時を放送日に対応する分だけ記憶させるような場合に比べて、記憶容量は小さくて済ませることが可能となる。

## 【 0 0 3 6 】

ポスティングデバイス (この例ではコントローラ 1 1) は、Scheduled Actionが所定の間隔 (intervals) で繰り返される場合、図 1 2 に示すようなフォーマットでイベントを記述する。この図 1 2 のrepeat\_typeには、この例の場合、図 1 0 に示す値" 1 0<sub>16</sub>" が記述される。number\_of\_eventsには、イベントの数



が記述される。interval（間隔）は、現在のイベントのstart\_timeから次のイベントのstart\_timeまでの間隔を表す。この間隔は、時分秒で表される。時間は3桁の時間を表す3つの数字がそれぞれ4ビットのBCDで表され、合計12ビットで表現される。分と秒は、それぞれ2桁の分と秒を表す2つの数字が4ビットのBCDとされ、それぞれ合計8ビットで表現される。このように、repeat\_typeには、周期的に繰り返されるイベントが記憶できるので、それぞれが開始される絶対時刻（日時）を別に記憶させる場合に比べて、記憶容量が少なくて済む。

## 【 0 0 3 7 】

さらに、図7に示すinfo Blocksは、図13に示すようなフォーマットとされる。

## 【 0 0 3 8 】

この図13において、compound\_lengthは、このinfo blockのバイト長を表す。ただしlengthフィールド自身は、この長さに含まれていない。info\_block\_typeは"8900<sub>16</sub>"にセットされる。primary\_fields\_lengthは、number\_of\_subunitsとsubunit\_type\_and\_ID fieldのバイト数を表す。number\_of\_subunitsは、ポスティングデバイス（この例ではコントローラ11）が使用するサブユニットの数を表す。subunit\_type\_and\_IDは、ポスティングデバイスが使用するサブユニットを指定する。

## 【 0 0 3 9 】

また、本実施の形態の場合、図7に示すResource Schedule Entryには、図14に示すcharacter\_code\_information\_blockと、図15に示すlanguage\_code\_information\_blockと、図16に示すraw\_text\_information\_blockとが記述される。

## 【 0 0 4 0 】

図14に示すcharacter\_code\_information\_blockには、図16のraw\_text\_information\_blockのraw\_text\_dataが何れのキャラクタコード（character code）でコーディングされているのかを示す情報が記述されている。この図14において、compound\_lengthは、このcharacter\_code\_information\_blockのバイト長を表す。ただし、lengthフィールド自身は、この長さに含まれていない。info\_blo

ck\_typeはcharacter\_code\_information\_blockであることを示す” 0 0 0 8<sub>16</sub>” にセットされる。primary\_fields\_lengthは、character\_code\_typeとcharacter\_code\_type\_specific fieldのバイト数を表す。character\_code\_typeはキャラクタコード (character code) の種類を表し、character\_code\_type\_specificはキャラクタコードの仕様を表す。

## 【 0 0 4 1 】

図 1 5 に示す language\_code\_information\_block には、図 1 6 の raw\_text\_information\_block の raw\_text\_data が何れの言語コード (language code) でコーディングされているのかを示す情報が記述されている。この図 1 5 において、compound\_length は、この language\_code\_information\_block のバイト長を表す。ただし、length フィールド自身は、この長さに含まれていない。info\_block\_type は language\_code\_information\_block であることを示す” 0 0 0 9<sub>16</sub>” にセットされる。primary\_fields\_length は、language\_code\_type と language\_code\_type\_specific field のバイト数を表す。language\_code\_type は言語コード (language code) の種類を表し、language\_code\_type\_specific は言語コードの仕様を表す。

## 【 0 0 4 2 】

図 1 6 に示す raw\_text\_information\_block において、compound\_length は、この raw\_text\_information\_block のバイト長を表す。ただし、length フィールド自身は、この長さに含まれていない。info\_block\_type は raw\_text\_information\_block であることを示す” 0 0 0 A<sub>16</sub>” にセットされる。primary\_fields\_length は、raw\_text\_data のバイト数を表す。raw\_text\_data には、図 1 7 に示すように、録画予約に関連する情報として、例えば、チャンネル、プログラム名 (番組名)、制御情報 (再生、録画、停止等を表すテキスト)、備考 (例えばペイパービューの番組等)、プロバイダ名、予約が仮予約である旨の情報などのテキストデータが記述される。

## 【 0 0 4 3 】

図 1 に示した本発明実施の形態のネットワークシステムにおいては、各ユニットの R S B の Resource Schedule Entry にテキスト情報を書き込むための information block として、character\_code\_information\_block と、language\_code\_info

rmation\_blockと、raw\_text\_information\_blockとを定義しており、これらを用いて、例えば図17に示したような録画予約に関連する情報など、機器の使用予定に関する様々な付加情報をテキストデータとして書き込み可能とし、また、このテキストデータはネットワークシステムを構成する各ユニットのコントローラにおいて読み取り可能となっている。

## 【0044】

したがって、本実施の形態のネットワークシステムにおいては、ユーザや各機器が上記録画予約に関連する情報を入力できること、各機器やユーザが他の機器による上記録画予約に関連する情報を入手できる（知り得る）こと、及び、各機器が他の機器やユーザに対して自己が持つ上記録画予約に関連する情報を知らせることなどが実現可能となっている。

## 【0045】

図18～図20には、本実施の形態のネットワークシステムにおいて上記録画予約に関連する情報の入力、入手、録画予約に関連する情報の報知を行う際の手順を示す。なお、これら図18～図20の処理は、録画予約に関連するリソース、すなわちサブユニットが複数存在する場合、それら各サブユニット毎に順番に行われる処理である。

## 【0046】

図18に示すフローチャートのステップS10からステップS17までは、使用したい機器のRSBが書き込み可能かどうかを調べる手順である。図1のネットワークシステムの例の場合、録画予約に使用するサブユニットは、IRD1のチューナサブユニット12とDVCR3のVCRサブユニット33であるため、IRD1のコントローラ11は、先ず、IRD1のBBS14内のRSBが書き込み可能であるかどうか調べる。

## 【0047】

この図18において、ユーザから例えば録画日時とチャンネルの録画予約の入力がなされると、IRD1のコントローラ11は、ステップS10の処理として、その予約の情報を例えば内部レジスタに一旦保存する。

## 【 0 0 4 8 】

コントローラ 1 1 は、上記予約設定入力となされると、次のステップ S 1 1 の処理として、上記予約設定の内容に応じて使用するサブユニットが存在する機器内の R S B をライトオープン (WRITE OPEN) させる処理 (すなわち書き込み可能な状態にする処理) を実行する。図 1 の例の場合、予約録画に使用するサブユニットは、I R D 1 のチューナサブユニット 1 2 と D V C R 3 の V C R サブユニット 3 3 であり、ここでは先ず、例えばチューナサブユニット 1 2 を有する機器 (ユニット) である I R D 1 の B B S 1 4 内の R S B をライトオープン (書き込み可能な状態) にする。なお、V C R サブユニット 3 3 を有する機器である D V C R 3 の B B S 3 4 の R S B 6 1 をライトオープンする処理は、上記 I R D 1 の R S B に対して図 1 8 ~ 図 2 0 の処理が行われた後になる。

## 【 0 0 4 9 】

ここで図 2 1 には、コントローラ 1 1 が R S B をライトオープンする場合に出力するライトオープンコマンド (WRITE OPEN コマンド) のフォーマットを示す。但し、I R D 1 の B B S 1 4 の R S B をライトオープンする場合、B B S 1 4 とコントローラ 1 1 とは、同じ I R D 1 内に設けられているものであり、その間にはバス 2 は存在していないので、この場合のコントローラ 1 1 は、B B S 1 4 に対して、バス 2 を介してライトオープンコマンドが供給されてきた場合と同様の状態となるように B B S 1 4 を制御する。なお、上記 I R D 1 の R S B に対して図 1 8 ~ 図 2 0 の処理が行われた後、D V C R 3 の B B S 3 4 の R S B 6 1 をライトオープンする際には、コントローラ 1 1 から図 2 1 に示すフォーマットのライトオープンコマンド (WRITE OPEN コマンド) が出力され、バス 2 を介して D V C R 3 の B B S 3 4 (実際には B B S 3 4 を制御するコントローラ 3 1) へ送られることになる。

## 【 0 0 5 0 】

この図 2 1 に示すライトオープンコマンドは、ターゲットの所定のアドレス空間にアクセス (この例では B B S の R S B にアクセス) するために使用される、オープンディスクリプタコマンド (OPEN DESCRIPTOR コマンド) の一種である。この図 2 1 に示すライトオープンコマンドにおいて、opcode にはオープンディス

クリプタであることを表す値”  $08_{16}$ ” が記述され、operand0にはライトオープンするディスクリプタの種類を表すdescriptor\_typeとして、リストIDにより規定されるObject List Descriptorであることを表す値”  $10_{16}$ ” が記述される。operand1とoperand2には、アクセス先の（ライトオープンする）RSBのリストID（この例においては、”00”と”01”）が記述される。さらにoperand3には、サブファンクション（subfunction）として、ディスクリプタを読み出しまたは書き込みアクセスのためにオープンするライトオープンであることを表す値”  $03_{16}$ ” が記述される。operand4は、リザーブのための値”00”とされている。

## 【0051】

次に、コントローラ11の処理は、ステップS12に進み、IRD1のBBS14におけるRSB内のdescriptor\_lengthとlist\_specific\_information fieldの情報を読み出す。なお、DVCR3のBBS34のRSB61の読み出し処理は、上記IRD1のRSBに対して図18～図20の処理が行われた後になる。

## 【0052】

図22には、コントローラ11がRSBをリード（READ）する場合に出力するリードコマンド（READコマンド）のフォーマットを示す。但し、IRD1のBBS14のRSBをリードする場合、BBS14とコントローラ11とは、同じIRD1内に設けられているものであり、その間にはバス2は存在していないので、この場合のコントローラ11は、BBS14に対して、バス2を介してリードコマンドが供給されてきた場合と同様の状態となるようにBBS14を制御する。なお、上記IRD1のRSBに対して図18～図20の処理が行われた後、DVCR3のBBS34のRSB61をリードオープンする際には、コントローラ11から図22に示すフォーマットのリードコマンド（READコマンド）が出力され、バス2を介してDVCR3のBBS34（実際にはBBS34を制御するコントローラ31）へ送られることになる。

## 【0053】

図22に示すリードコマンドのフォーマットにおいて、先頭のopcodeには、リードディスクリプタ（read descriptor）であることを表す値の”  $09_{16}$ ” が記

述されている。続くoperand 0には読み出すディスクリプタ(descriptor)を識別するためのdescriptor identifierが記述される。上記ステップS 1 2における読み出しの処理の場合は、リストIDでdescriptor identifierが記述される。具体的には前記図4に示したRSBのwrite enabled list\_specific\_information fieldのAddress\_offsetの"00<sub>16</sub>"乃至"0D<sub>16</sub>"が記述される。read\_result\_statusには、ポスティングデバイスがリードコマンドを送出するときは"FF<sub>16</sub>"が記述され、ターゲット(この例ではBBSのRSB)からレスポンスとして返される場合には、読み取り結果が記述される。data\_lengthには、ターゲットから読み出されるべきデータのバイト数が記述される。この値が"00<sub>16</sub>"に設定された時、全てのリストが読み出される。addressには、読み出しを開始すべきアドレスが記述される。その値が"00<sub>16</sub>"とされた場合、先頭から読み出しが開始される。

## 【0054】

次に、コントローラ11は、ステップS 1 3の処理として、この時点でのターゲットであるIRD1のBBS14内のRSBに対するリストの最大長の制限(図4におけるobject\_list\_maximum\_size)、リストのエントリ数の制限(図4におけるobject\_entries\_maximum\_number)、および各エントリの最大バイト長の制限(図4におけるobject\_entry\_maximum\_size)を抽出する。

## 【0055】

次に、コントローラ11は、ステップS 1 4の処理として、これからIRD1のBBS14内のRSBにデータ(この場合予約情報)を記録しても、ステップS 1 3で抽出されたリストの最大長(object\_list\_maximum\_size)を超えないか否かを判定する。ステップS 1 4において上記リストの最大長(object\_list\_maximum\_size)を越えないと判定した場合、コントローラ11の処理はステップS 1 5に進み、一方、越えていると判定した場合、コントローラ11の処理はステップS 1 7の処理に進む。

## 【0056】

ステップS 1 5の処理に進むと、コントローラ11は、ステップS 1 3で抽出されたリストのエントリ数の制限(最大エントリ数: object\_entries\_maximum\_n



umber) から、現在のエントリ数を引いた値が” 0 ” より大きいかな否か、すなわち、まだ記録可能なエントリが残っているかな否かを判定する。当該ステップ S 1 5 において大きい（記録可能なエントリが残っている）と判定された場合、コントローラ 1 1 の処理はステップ S 1 6 に進み、一方、大きくない（記録可能なエントリが残っていない）と判定された場合、コントローラ 1 1 の処理はステップ S 1 7 の処理に進む。

#### 【 0 0 5 7 】

ステップ S 1 6 の処理に進むと、コントローラ 1 1 は、ステップ S 1 3 で抽出された最大エントリ長 (object\_entry\_maximum\_size) から、これから書き込もうとしている予約情報のエントリ長を引いた値が” 0 ” より大きいかな否か、すなわち、書き込むべきエントリ長に、まだ余裕があるかな否かを判定する。当該ステップ S 1 6 において大きい（エントリ長に余裕がある）と判定された場合、コントローラ 1 1 の処理は図 1 9 のステップ S 1 8 に進み、大きくない（エントリ長に余裕がない）と判定された場合、コントローラ 1 1 の処理はステップ S 1 7 に進む。

#### 【 0 0 5 8 】

ステップ S 1 4 乃至ステップ S 1 6 における条件の何れか 1 つが満足されずにステップ S 1 7 の処理に進むと、コントローラ 1 1 は、例えば「予約が一杯です」のような警告表示を図示しない表示手段上に行う。すなわち、この場合の警告表示は、IRD 1 の BBS 1 4 内の RSB に予約情報を書き込むだけの余裕がなくなっていることを表している。これにより、ユーザは、予約が一杯でそれ以上予約をすることができないことを知ることができる。また、IRD 1 の BBS 1 4 内の RSB の図 7 に示した Resource Schedule Entry に、前述の図 1 7 に示した予約の詳細な内容を表すテキスト情報が既に記述されている場合には、そのテキスト情報の表示を行う。これにより、ユーザは、その予約の詳細な内容を知ることができる。本実施の形態において上述のように予約の詳細な内容を例えばテキスト表示することにより、ユーザは、例えばそれら予約が必要な予約かそうでないかな等の判断、すなわち必ず録画したい番組であるか或いは必ずしも録画しなくてもよい番組であるかな等の判断ができ、例えば必ず録画したい番組についての



予約はそのまま維持させ、一方、必ずしも録画しなくてもよい番組についてはその予約を取り消し、別の番組の予約を新たに行うなどの、適切な行動を取ることができるようになる。また例えば、予約が仮予約なので上書きしても良いという情報をテキスト情報としてエントリ時に書き込むようにしておけば、後から予約するユーザがより適切な行動、すなわち仮予約されていたものを消して新たな予約を行ったりする等の行動を取ることができるようになる。

## 【 0 0 5 9 】

一方、ステップ S 1 4 乃至ステップ S 1 6 の条件の何れもが満足されている場合、I R D 1 の B B S 1 4 内の R S B に予約情報を書き込む余裕があるので、コントローラ 1 1 は、図 1 9 のステップ S 1 8 以降の処理に進み、重複する時刻の予約が既になされているか否かの判定を行う。

## 【 0 0 6 0 】

すなわち、コントローラ 1 1 は、先ず図 1 9 のステップ S 1 8 の処理として、変数  $i$  を " 0 " に初期設定し、次にステップ S 1 9 の処理として、I R D 1 の B B S 1 4 の R S B に記録されているエントリの数 (number\_of\_entries) から変数  $i$  を減算した値が " 0 " より大きいかな否かを、すなわち B B S 1 4 の R S B に記録されている全てのエントリについて検索が行われたかな否かを判定する。このステップ S 1 9 の処理において、number\_of\_entries から変数  $i$  を減算した値が " 0 " より大きいと判定した場合、まだ検索していないエントリが存在するので、コントローラ 1 1 の処理はステップ S 2 0 に進み、B B S 1 4 の R S B に掲示されている前記図 6 の object\_entry [  $i$  ] を読み出す（この場合は object\_entry [ 0 ] を読み出す）。なお、当該ステップ S 2 0 における読み出しも、前記図 2 2 に示したリードコマンドで行われるが、この場合における descriptor identifier の記述は object position で行われる。この object entry [  $i$  ] には、既に登録されている予約の時刻情報（図 7 における start\_time, Duration）や、その予約において、使用するサブユニットの識別情報（図 1 3 における subunit\_type\_and\_ID [ 0 ] ）、前記図 1 4 ～図 1 7 に示したテキスト情報などが記憶されている。

## 【 0 0 6 1 】

次に、コントローラ 1 1 は、ステップ S 2 1 の処理として、ステップ S 1 0 でユーザより入力された時刻情報 (start\_time, Duration) が、ステップ S 2 0 で読み出された時刻情報 (start\_time, Duration) と重複しているか否かを判定する。このステップ S 2 1 において、時刻が重複していると判定された場合、コントローラ 1 1 はステップ S 2 2 の処理に進み、一方、重複していないと判定した場合、コントローラ 1 1 はステップ S 2 3 の処理に進む。

## 【 0 0 6 2 】

ステップ S 2 1 にて時刻が重複していると判定してステップ S 2 2 の処理に進むと、コントローラ 1 1 は、ステップ S 1 0 で予約設定されたサブユニット（この場合、チューナサブユニット 1 2）が、ステップ S 2 0 で読み出したサブユニット (subunit\_type\_and\_ID) と一致しているか否かを判定する。当該ステップ S 2 2 にてサブユニットが一致していると判定した場合、コントローラ 1 1 の処理はステップ S 2 4 に進み、一方、サブユニットが一致していないと判定した場合、コントローラ 1 1 の処理はステップ S 2 3 に進む。

## 【 0 0 6 3 】

ステップ S 2 2 にてサブユニットが一致していると判定された場合は、結局、時刻とサブユニットの両方が一致していることになるので、コントローラ 1 1 はステップ S 2 4 の処理として、例えば「予約が重なっています」のような警告表示を行う。これにより、ユーザは、予約が重なることを知ることができて予約のダブルブッキングの発生を防止できることになる。また、IRD 1 の BBS 1 4 のRSBの図 7 に示した Resource Schedule Entry に、テキスト情報として予約の設定内容が記述されている場合は、そのテキスト情報の表示を行う。これにより、ユーザは、その重なる予約の詳細な内容を知ることができる。本実施の形態において予約の詳細な内容を例えばテキスト表示することにより、ユーザは、例えばそれら予約が必要な予約かそうでないか等の判断ができ、例えば必要な予約についてはそのまま維持させ、また必ずしも必要でない予約を取り消し、別の新たな予約を行うなどの適切な行動を取ることができるようになる。また例えば、予約が仮予約なので上書きしても良いという情報をテキスト情報としてエントリ

時に書き込むようにしておけば、後から予約するユーザがより適切な行動、すなわち仮予約されていたものを消して新たな予約を行ったりする等の行動を取ることができるようになる。

## 【 0 0 6 4 】

一方で、ステップ S 2 1 において、時刻が重複していないと判定された場合、又は、時刻が一致していたとしてもステップ S 2 2 においてサブユニットが一致していないと判定された場合、予約のダブルブッキングが発生する恐れはないので、コントローラ 1 1 は、ステップ S 2 3 の処理として、変数 *i* を 1 だけインクリメントした後、ステップ S 1 9 に戻り、`number_of_entries` から変数 *i* を減算した値が 0 より大きくないと判定されるまで、同様の処理を繰り返し実行する。すなわち、コントローラ 1 1 では、IRD 1 の BBS 1 4 の RSB に記憶されている全ての `object entry [i]` について、重複する時刻の予約がなされているか否かの検索を行う。

## 【 0 0 6 5 】

また、ステップ S 1 9 において、`number_of_entries` から変数 *i* を減算した値が” 0 ” より大きくないと判定された場合（全ての `object entry [i]` の検索が終了した場合）、コントローラ 1 1 の処理は、図 2 0 のステップ S 2 5 に進む。なお、前述したステップ S 1 4 乃至ステップ S 1 7 の処理は、ステップ S 1 9 の処理において、NO と判定された場合に実行するようにすることも可能である。

## 【 0 0 6 6 】

図 2 0 のステップ S 2 5 の処理に進むと、コントローラ 1 1 は、IRD 1 の BBS 1 4 の RSB の `object entry` をクリエイト (create) する。なお、DVCR 3 の BBS 3 4 の RSB のクリエイト処理は、上記 IRD 1 の RSB に対して図 1 8 ～図 2 0 の処理が行われた後になる。

## 【 0 0 6 7 】

図 2 3 には、コントローラ 1 1 が RSB の `object entry` をクリエイト (create) する場合に出力するクリエイトコマンド (CREATE コマンド) のフォーマットを示す。但し、IRD 1 の BBS 1 4 の RSB をクリエイトする場合、BBS 1 4 とコントローラ 1 1 とは、同じ IRD 1 内に設けられているものであり、その間

にはバス 2 は存在していないので、この場合のコントローラ 1 1 は、B B S 1 4 に対して、バス 2 を介してクリエイトコマンドが供給されてきた場合と同様の状態となるように B B S 1 4 を制御する。なお、上記 I R D 1 の R S B に対して図 1 8 ～図 2 0 の処理が行われた後、D V C R 3 の B B S 3 4 の R S B 6 1 の object entry をクリエイトする際には、コントローラ 1 1 から図 2 3 に示すフォーマットのクリエイトコマンド (CREATE コマンド) が出力され、バス 2 を介して D V C R 3 の B B S 3 4 (実際には B B S 3 4 を制御するコントローラ 3 1) へ送られることになる。

【 0 0 6 8 】

また、図 2 4 には、図 2 3 内の subfunction\_1 で指定できる値を示し、本実施の形態では " 0 1 " (create a new object and its child list) が使用される。また、図 2 5 は、図 2 3 内の subfunction\_1\_specification for subfunction\_1 = 0 1 のフォーマットを示す。さらに図 2 6 は、図 2 5 内の各フィールド値を示した図である。図 2 5 の descriptor\_identifier\_where, descriptor\_identifier\_what\_1, \_2 の各フィールドに、図 2 6 に示すように、" 2 0<sub>16</sub>" , " 2 2<sub>16</sub>" , " 1 1<sub>16</sub>" をそれぞれ設定すると、" create a new object and its child list" の意味となる。

【 0 0 6 9 】

これら A V / C コマンドにおけるクリエイトコマンドについての詳細は、I E E E 1 3 9 4 (インターネットホームページ <http://www.1394TA.org> 参照) に記述されているものであり、本実施の形態中の各図は、その文献 (Enhancement to the AV/C General Specification 3.0 Version 1.0 FC2 や、TA Document 19990 05 AV/C Bulletin Board Subunit General Specification 1.0 Draft 0.99:149) 中のものを記載してある。また、ボードを構成するインフォメーションディスクリプタ (Information List Descriptor) にも書き込み可能なものと読み出し可能なものがあり、これらの区別には、リストタイプが使用される。

【 0 0 7 0 】

なお、外部から A V / C ディスクリプタ (AV/C Descriptor) に新規に情報を書き込む方法の 1 つとして、例えばコントローラ (ポスティングデバイス) がタ

ターゲットに対して前述したクリエイトコマンドを発行し、当該ターゲットが情報を書き込む雛形を作った後、再度、コントローラが具体的な内容を書き込む制御を行うような方法が一般的な方法として考えられる。例えば、初めて情報を書き込む場合、コントローラは所望のリストを指定して、AV/Cディスクリプタクリエイトコマンド (AV/C Descriptor CREATE command) を発行する。このコマンドを受けたターゲットでは、AV/Cジェネラルで指定されたデータ構造の雛形に基づいたオブジェクトをターゲットの内部に作ることになる。また、AV/Cジェネラルで決められたデータ構造の雛形には、オブジェクトIDを示すフィールドがある。AV/Cディスクリプタ (AV/C Descriptor) を用いたリストでは、オブジェクトIDはターゲットが管理することになる。つまり、オブジェクトをクリエイト (CREATE) した段階で、ターゲットがそのオブジェクトを一意に指定できるIDを付け、そのIDを管理する機能をターゲットが所有することになる。

#### 【 0 0 7 1 】

オブジェクトIDとは、リスト内でそのオブジェクトを一意に指定するための識別番号であり、このため当該オブジェクトIDを重複しないようにする機能が管理する側に必要になる。BBS自体は情報を提供する場所であり、オブジェクトIDの管理はコントローラが持つことになる。

#### 【 0 0 7 2 】

ところが、サブユニットに対してクリエイトコマンドが発行された時、矛盾が生ずる虞がある。すなわち、オブジェクトをクリエイト (CREATE) した際には、コントローラが管理すべきオブジェクトIDを、ターゲットが割り振ることになるからである。また、クリエイトコマンドの発行後は、ライト制御を続けて行う必要がある。このように、処理が複数ステップにわかれていることにより、コントローラが書き込み途中で例えばバスから外されたような場合には、不完全なオブジェクトが作成されてしまう可能性がある。

#### 【 0 0 7 3 】

したがって、上記のような状況においては、その不完全なオブジェクトを特定し、そのようなオブジェクトができたときに、当該オブジェクトを良好に削除で

きるシステムが必要になる。

【 0 0 7 4 】

そこで、本発明の実施の形態では、B B S への書き込み手段を規格で規定し、不完全なオブジェクトを特定できる仕組みを用意している。

【 0 0 7 5 】

すなわち、まず、ターゲットは、オブジェクト I D (グローバルユニーク I D (Grobal Unique ID : GUID) と record ID とで構成される) のうちの G U I D の部分に、一時的に管理する番号 (例えば、全て " 0 " ) を割り振るようにする。コントローラは、先にオブジェクト内部に情報を書き込み、正常に書き込みが終了したならば、最後に G U I D を書き換える。

【 0 0 7 6 】

上記の手順を決めることにより、正常に書き込み作業が終了したときには、G U I D が全て " 0 " となっているオブジェクトはできないことになり、したがって、G U I D が全て " 0 " のオブジェクトは、書き込み途中で不完全となったオブジェクトと特定できることになる。

【 0 0 7 7 】

これにより、書き込み途中のオブジェクトを一意に特定することができ、また、正常に書き込まれたオブジェクトと不完全なオブジェクトとを区別でき、さらに不完全なオブジェクト (無効なオブジェクト) を簡単に削除することが可能となる。このことにより、電子機器に設けられている有限なメモリを有効活用できるようになる。また、書き込み途中のオブジェクトの特定方法は、オブジェクト I D の G U I D 部分を全て " 0 " にするような簡単な方法なので、不完全なオブジェクトを削除するためのソフトウェアの作成も容易となる。

【 0 0 7 8 】

なお、図 2 0 のステップ S 2 5 において、上記クリエイトコマンドを利用する代わりに、インサートコマンド (INSERT コマンド) を使用することも可能である。

【 0 0 7 9 】

次に、ステップ S 2 6 の処理に進むと、コントローラ 1 1 は、I R D 1 の B B



S 1 4 の R S B の entry\_specific\_information fields の部分 (図 2、図 7) に予約内容を書き込む。すなわち、これにより、start\_time, Duration, repeat\_information, 使用するサブユニット (subunit\_type\_and\_id) などが書き込まれる。

#### 【 0 0 8 0 】

図 2 7 は、このような場合にコントローラ 1 1 が出力するライトディスクリプタコマンド (WRITE DESCRIPTOR コマンド) のフォーマットを表している。但し、I R D 1 の B B S 1 4 の R S B に書き込みを行う場合、B B S 1 4 とコントローラ 1 1 とは同じ I R D 1 内に設けられているため、この場合のコントローラ 1 1 は、B B S 1 4 に対して、バス 2 を介してライトディスクリプタコマンドが供給されてきた場合と同様の状態となるように B B S 1 4 を制御する。なお、上記 I R D 1 の R S B に対して図 1 8 ~ 図 2 0 の処理が行われた後、D V C R 3 の B B S 3 4 の R S B 6 1 に書き込みを行う際には、コントローラ 1 1 から図 2 7 に示すフォーマットのライトディスクリプタコマンドが出力され、バス 2 を介して D V C R 3 の B B S 3 4 (実際には B B S 3 4 を制御するコントローラ 3 1) へ送られることになる。

#### 【 0 0 8 1 】

図 2 7 において、先頭の opcode には、WRITE DESCRIPTOR であることを表す値 "0 A<sub>16</sub>" が記述される。operand 0 には、書き込み対象となるディスクリプタを識別させるための descriptor identifier が記述される。この記述は、object position で行われる。以下、subfunction として、partial\_replace であることを表す値 "5 0<sub>16</sub>" が記述される。これによりパーシャルインサート (partial insert) またはパーシャルディレート (partial delete) が実行される。インサート (insert) では、descriptor\_identifier で指定される operand により規定される 1 つ前に新しいディスクリプタ (descriptor) が挿入される。ディレート (delete) では descriptor\_identifier により規定されるディスクリプタが削除される。group\_tag は、ライトディスクリプタコマンドが発行される必要があるディスクリプタ上において、分割できない更新の処理を行うために利用される。この例においては、データをディスクリプタに迅速に書き込むことを表す値 "0 0<sub>16</sub>"



” (immediate) が記述されている。replacement\_data\_lengthは、replacement\_dataのoperandにおけるバイト数、すなわち書き込みたいデータの長さを表している。addressは、処理が行われるべき位置を表している。replacement\_data\_lengthの” 0 ” はパーシャルデリート (partial delete) を意味し、その場合、replacement\_dataのoperandは存在しない。この場合、originalにdata\_lengthは” 0 ” より大きい値となり、それが削除されるべきバイトの数を表す。original\_data\_lengthが” 0 ” である場合、パーシャルインサート (partial insert) 処理が行われる。この場合、replacement\_data\_lengthは、” 0 ” より大きい値とされ、挿入されるべきバイトの数を表している。

## 【 0 0 8 2 】

次に、コントローラ 1 1 は、ステップ S 2 7 の処理として、IRD 1 の B B S 1 4 内の R S B の図 7 に示した Resource Schedule Entryに、テキスト情報として予約の設定内容を追加 (記述) するか否かを判定し、追加すると判定した場合はステップ S 2 8 の処理に進み、追加しないと判定した場合はステップ S 3 1 の処理に進む。

## 【 0 0 8 3 】

ステップ S 2 8 に進むと、コントローラ 1 1 は、先ず図 2 8 に示す Subunit Identifier Descriptor内の R S B ボードタイプを読み出す。

## 【 0 0 8 4 】

図 2 8 において、descriptor\_Lengthは、このディスクリプタ (Descriptor) の長さを表す。generation\_IDは、この B B S において、どの A V / C ディスクリプタ (AV/C descriptor) フォーマットが使用されるかを表し、通常” 0 0<sub>16</sub> ” とされる。size\_of\_list\_IDは、list IDのバイト数を表す。size\_of\_object\_IDは、このobject IDのバイト数を表す。size\_of\_object\_posionは、list内のオブジェクト (object) の位置が参照されるときに使用されるバイト数を表す。number\_of\_root\_object\_listsは、この B B S が直接関連するroot object listsの数を表す。root\_object\_list\_id\_x (x= 0, 1, 2, . . . , n-1) は、この B B S が関連するroot object listsのそれぞれの I Dを表す。subunit\_dependent\_information\_lengthはsubunit\_dependent\_informationの長さを表し、subuni

t\_dependent\_informationには、このBBSが依存するフォーマット及びコンテンツに関する情報が記述される。subunit\_dependent\_informationには、non\_info\_blockにfields\_length, bulletin\_board\_subunit\_version, number\_of\_supported\_board\_type (n) , supported\_board\_type\_specific\_of\_length [0] の他、supported\_board\_type\_specific\_info [0] 乃至supported\_board\_type\_specific\_info [n-1] と、それらの長さを表すsupported\_board\_type\_specific\_of\_length [0] 乃至supported\_board\_type\_specific\_of\_length [n-1] が含まれている。さらに、BBSには、manufacturer\_dependent\_informationの長さを表す、manufacturer\_dependent\_lengthと製造者に依存する情報を含むmanufacturer\_dependent\_informationが記述される。root\_object\_list\_idの値は、それがRSBを表すものである場合、図29に示すように、所定の値”1001<sub>16</sub>”とされる。このように、RSB（図29ではResource Schedule Listと記述されている）を表すIDを、所定の値に固定しておくことにより、RSBを読み出す処理が容易となる。

## 【0085】

また、図28のsupported\_board\_type\_specific\_informationフィールドは、図30に示すようなフォーマットとされる。この図30において、supported\_board\_typeには、図5に示したRSBであることを表す値”01<sub>16</sub>”が記述される。supported\_board\_type\_versionは、bulletin Board Type Specificationのバージョンの番号を表す。implementation\_profile\_IDはこのboard typeのためのprofile IDバージョンを表す。supported\_board\_type\_dependent\_information\_lengthは、supported\_type\_dependent\_informationのバイト数を表す。supported\_board\_type\_dependent\_informationには、各board type specificationに固有の情報が記述される。

## 【0086】

次に、ステップS29の処理に進み、コントローラ11は、BBS14のRSBのバージョンが1.0、すなわちRSBのResource Schedule Entryにテキスト情報を書き込むためのinformation blockとして、前記図7及び図14～図17に示したcharacter\_code\_information\_blockと、language\_code\_information\_

blockと、raw\_text\_information\_blockとが定義されているバージョン1.0よりも、RSBのバージョンが大きいかな否かを判定し、大きいと判定したときはステップS30の処理に進み、大きくないと判定したときはステップS31の処理に進む。

## 【0087】

ステップS30の処理に進むと、コントローラ11は、ユーザからの予約設定情報の入力に応じたテキスト情報を、上記BBS14のRSBのResource Schedule Entry内のテキスト情報を書き込むためのinformation blockに追加する。

## 【0088】

次に、ステップS31の処理に進むと、コントローラ11は、リスト、すなわちBBS14のRSBをクローズする。図31には、RSBをクローズする場合にコントローラ11が出力するクローズコマンド（CLOSEコマンド）のフォーマットを表している。但し、IRD1のBBS14のRSBをクローズする場合、BBS14とコントローラ11とは同じIRD1内に設けられているため、この場合のコントローラ11は、BBS14に対して、バス2を介してクローズコマンドが供給されてきた場合と同様の状態となるようにBBS14を制御する。なお、上記IRD1のRSBに対して図18～図20の処理が行われた後、DVCR3のBBS34のRSBのクローズを行う際には、コントローラ11から図31に示すフォーマットのクローズコマンドが出力され、バス2を介してDVCR3のBBS34（実際にはBBS34を制御するコントローラ31）へ送られることになる。

## 【0089】

この図31に示すクローズコマンドのフォーマットは、基本的に図21に示したライトオープンコマンドと同様のフォーマットであり、subfunctionが、図21においては、ライトオープン（WRITE OPEN）を表す”03<sub>16</sub>”とされているのに対して、図31のクローズコマンドにおいては、クローズ（CLOSE）であることを表す値”00<sub>16</sub>”とされている点が異なっている。その他の構成は図21における場合と同様である。

## 【 0 0 9 0 】

次にステップ S 3 2 に進み、コントローラ 1 1 は、予約に関連する他のリソースが存在するか否かを判定する。この場合、これまでの処理により I R D 1 のチューナサブユニット 1 2 についての予約の処理は終了しているが、D V C R 3 の V C R サブユニット 3 3 についての予約の処理は行われていないため、当該ステップ S 3 2 において他のリソースがあると判定され、図 1 8 のステップ S 1 1 の処理に戻る。

## 【 0 0 9 1 】

これ以降、コントローラ 1 1 は、当該他のリソースとして D V C R 3 の V C R サブユニット 3 3 についての予約の処理を実行する。すなわち、コントローラ 1 1 は、当該 D V C R 3 の S S B 3 4 の R S B 6 1 に対して、前述同様に、ステップ S 1 1 以降の処理を行うことになる。

## 【 0 0 9 2 】

当該 D V C R 3 の V C R サブユニット 3 3 についての予約処理を実行する場合のコントローラ 1 1 は、ステップ S 1 1 の処理として、V C R サブユニット 3 3 を有する機器である D V C R 3 の B B S 3 4 内の R S B 6 1 を、ライトオープン（書き込み可能な状態）するために、前記図 2 1 に示したライトオープンコマンドを、バス 2 を介して D V C R 3 の B B S 3 4 （実際には B B S 3 4 を制御するコントローラ 3 1）へ送る。

## 【 0 0 9 3 】

次に、コントローラ 1 1 は、ステップ S 1 2 の処理として、前記図 2 2 に示したリードコマンドをバス 2 を介して D V C R 3 の B B S 3 4 （実際には B B S 3 4 を制御するコントローラ 3 1）へ送り、R S B 6 1 の descriptor\_length と list\_specific\_information field の情報を読み出す。

## 【 0 0 9 4 】

次に、コントローラ 1 1 は、ステップ S 1 3 の処理として、D V C R 3 の B B S 3 4 内の R S B 6 1 に対するリストの最大長の制限（図 4 における object\_list\_maximum\_size）、リストのエントリ数の制限（図 4 における object\_entries\_maximum\_number）、および各エントリの最大バイト長の制限（図 4 における objec

t\_entry\_maximum\_size) を抽出する。

【 0 0 9 5 】

次に、コントローラ 1 1 は、前記ステップ S 1 4 ～ステップ S 1 6 の処理を行う。ステップ S 1 4 乃至ステップ S 1 6 における条件の何れか 1 つが満足されずにステップ S 1 7 の処理に進むと、コントローラ 1 1 は、例えば「予約が一杯です」のような警告表示を図示しない表示手段上に行う。すなわち、この場合の警告表示は、DVCR 3 の BBS 3 4 内の RSB 6 1 に予約情報を書き込むだけの余裕がなくなっていることを表している。これにより、ユーザは、予約が一杯でそれ以上予約をすることができないことを知ることができる。また、DVCR 3 の BBS 3 4 の RSB 6 1 の図 7 に示した Resource Schedule Entry に、前述の図 1 7 に示した予約の詳細な内容を表すテキスト情報が既に記述されている場合には、そのテキスト情報の表示を行う。これにより、ユーザは、その予約の詳細な内容を知ることができる。本実施の形態において上述のように予約の詳細な内容を例えばテキスト表示することにより、ユーザは、例えばそれら予約が必要な予約かそうでないか等の判断、すなわち必ず録画したい番組であるか或いは必ずしも録画しなくてもよい番組であるか等の判断ができ、例えば必ず録画したい番組についての予約はそのまま維持させ、一方、必ずしも録画しなくてもよい番組についてはその予約を取り消し、別の番組の予約を新たに行うなどの、適切な行動を取ることができるようになる。また例えば、予約が仮予約なので上書きしても良いという情報をテキスト情報としてエントリ時に書き込むようにしておけば、後から予約するユーザがより適切な行動、すなわち仮予約されていたものを消して新たな予約を行ったりする等の行動を取ることができるようになる。

【 0 0 9 6 】

一方、ステップ S 1 4 乃至ステップ S 1 6 の条件の何れもが満足されている場合、DVCR 3 の BBS 3 4 内の RSB 6 1 に予約情報を書き込む余裕があるので、コントローラ 1 1 は、図 1 9 のステップ S 1 8 以降の処理に進み、重複する時刻の予約が既になされているか否かの判定を行う。すなわち、コントローラ 1 1 は、図 1 9 のステップ S 1 8 の処理として変数 i を初期設定し、次にステップ S 1 9 の処理として、BBS 3 4 の RSB 6 1 に記録されているエントリの数（

number\_of\_entries) から変数 *i* を減算した値が” 0 ” より大きいかな否かを判定する。このステップ S 1 9 の処理において、number\_of\_entries から変数 *i* を減算した値が” 0 ” より大きいと判定した場合、まだ検索していないエントリが存在するので、コントローラ 1 1 はステップ S 2 0 の処理として、DVCR 3 の BBS 3 4 の RSB 6 1 に掲示されている前記図 6 の object\_entry [ *i* ] を読み出す。

#### 【 0 0 9 7 】

次に、コントローラ 1 1 は、ステップ S 2 1 の処理として、ステップ S 1 0 でユーザより入力された時刻情報 (start\_time, Duration) が、ステップ S 2 0 で読み出された時刻情報 (start\_time, Duration) と重複しているかな否かを判定する。

#### 【 0 0 9 8 】

ステップ S 2 1 にて時刻が重複していると判定してステップ S 2 2 の処理に進むと、コントローラ 1 1 は、ステップ S 1 0 で予約設定されたサブユニット（この場合、VCR サブユニット 3 3）が、ステップ S 2 0 で読み出したサブユニット (subunit\_type\_and\_ID) と一致しているかな否かを判定する。

#### 【 0 0 9 9 】

ステップ S 2 2 にてサブユニットが一致していると判定された場合は、結局、時刻とサブユニットの両方が一致することになるので、コントローラ 1 1 はステップ S 2 4 の処理として、例えば「予約が重なっています」のような警告表示を行う。これにより、ユーザは、予約が重なることを知ることができて予約のダブルブッキングの発生を防止できることになる。また、DVCR 3 の BBS 3 4 の RSB 6 1 の図 7 に示した Resource Schedule Entry に、テキスト情報として予約の設定内容が記述されている場合は、そのテキスト情報の表示を行う。これにより、ユーザは、その重なる予約の詳細な内容を知ることができる。本実施の形態において予約の詳細な内容を例えばテキスト表示することにより、ユーザは、例えばそれら予約が必要な予約かそうでないか等の判断ができ、例えば必要な予約についてはそのまま維持させ、また必ずしも必要でない予約を取り消し、別の新たな予約を行うなどの適切な行動を取ることができるようになる。また例



例えば、予約が仮予約なので上書きしても良いという情報をテキスト情報としてエントリ時に書き込むようにしておけば、後から予約するユーザがより適切な行動、すなわち仮予約されていたものを消して新たな予約を行ったりする等の行動を取ることができるようになる。

#### 【 0 1 0 0 】

一方で、ステップ S 2 1 において、時刻が重複していないと判定された場合、又は、時刻が一致していたとしてもステップ S 2 2 においてサブユニットが一致していないと判定された場合、予約のダブルブッキングが発生する恐れはないので、コントローラ 1 1 は、ステップ S 2 3 の処理として、変数 *i* を 1 インクリメントした後、ステップ S 1 9 に戻り、number\_of\_entries から変数 *i* を減算した値が 0 より大きくないと判定されるまで、同様の処理を繰り返し実行する。すなわち、コントローラ 1 1 では、DVCR 3 BBS 3 4 の RSB 6 1 に記憶されている全ての object entry [*i*] について、重複する時刻の予約がなされているか否かの検索を行う。

#### 【 0 1 0 1 】

また、ステップ S 1 9 において、number\_of\_entries から変数 *i* を減算した値が " 0 " より大きくないと判定された場合（全ての object entry [*i*] の検索が終了した場合）、コントローラ 1 1 は、図 2 0 のステップ S 2 5 の処理として、前記図 2 3 に示したクリエイトコマンドをバス 2 を介して DVCR 3 の BBS 3 4（実際には BBS 3 4 を制御するコントローラ 3 1）へ送り、当該 BBS 3 4 の RSB 6 1 の object entry をクリエイトする。

#### 【 0 1 0 2 】

次に、コントローラ 1 1 は、ステップ S 2 6 の処理として、前記図 2 7 に示したライトディスクリプタコマンドをバス 2 を介して DVCR 3 の BBS 3 4（実際には BBS 3 4 を制御するコントローラ 3 1）へ送り、当該 DVCR 3 の BBS 3 4 の RSB 6 1 の entry\_specific\_information fields の部分（図 2、図 7）に予約内容を書き込む。これにより、start\_time, Duration, repeat\_information, 使用するサブユニット（subunit\_type\_and\_id）などが書き込まれる。



## 【 0 1 0 3 】

次に、コントローラ 1 1 は、ステップ S 2 7 の判定処理において、D V C R 3 の B B S 3 4 内の R S B 6 1 の図 7 に示した Resource Schedule Entry に、テキスト情報として予約の設定内容を追加すると判定した場合、ステップ S 2 8 の処理として、図 2 8 に示した Subunit Identifier Descriptor 内の R S B ボードタイプを読み出し、次のステップ S 2 9 にて、B B S 3 4 の R S B 6 1 のバージョンが 1. 0、すなわち R S B の Resource Schedule Entry にテキスト情報を書き込むための information block として、図 7 及び図 1 4 ～図 1 7 に示した character\_code\_information\_block と、language\_code\_information\_block と、raw\_text\_information\_block とが定義されているバージョン 1. 0 よりも、R S B のバージョンが大きいかなかを判定する。

## 【 0 1 0 4 】

ステップ S 2 9 にてバージョン 1. 0 より大きいと判定してステップ S 3 0 に進むと、コントローラ 1 1 は、ユーザからの予約設定情報の入力に応じたテキスト情報を、上記 R S B 6 1 の Resource Schedule Entry の information block に追加する。

## 【 0 1 0 5 】

次に、ステップ S 3 1 の処理に進むと、コントローラ 1 1 は、図 3 1 に示したクローズコマンドをバス 2 を介して D V C R 3 の B B S 3 4 （実際には B B S 3 4 を制御するコントローラ 3 1）へ送り、当該 B B S 3 4 の R S B 6 1 をクローズする。

## 【 0 1 0 6 】

その後ステップ S 3 2 に進み、コントローラ 1 1 は、予約に関連する他のリソースが存在するか否かを判定する。この場合、これまでの処理により I R D 1 のチューナサブユニット 1 2 と D V C R 3 の V C R サブユニット 3 3 についての予約の処理は終了しており、他に予約処理を行うべきサブユニットがないため、処理を終了する。

## 【 0 1 0 7 】

次に、オブジェクト I D 設定処理について、図 3 2 のフローチャートを参照し

て説明する。この処理は、ユーザが I R D 1 に対して、D V C R 3 を用いる録画予約の操作を入力し、その操作がコントローラ 1 1 に検知され、上述した図 1 8 乃至図 2 0 の予約処理が実行されて、その録画予約が認められた後、開始される。この例においては、所定の時刻から所定の時間の間、実行されるイベント（予約処理）に関し、それを識別するために、オブジェクト I D が対応される。このオブジェクト I D は、7 2 ビットで構成され、そのうちの M S B 側の 6 4 ビットは、その予約に関する全ての情報を持っている機器の固有の I D、具体的にはその機器の G U I D (Global Unique ID) とされ、L S B 側の 8 ビットは、その機器内で設定される固有の値とされ、具体的にはレコード I D (record ID) とされる。このようにオブジェクト I D を G U I D とレコード I D とで構成することにより、イベントを識別するためのオブジェクト I D を設定すると、I D 設定の処理が容易となる。

#### 【 0 1 0 8 】

すなわちこのオブジェクト I D は、バス 2 に接続されている各機器の間において、識別可能である必要がある。そうでなければ、この例においては、1 つのユニット（機器）の R S B の内容を他のユニット（機器）が読み込み可能であり、複数のユニットのサブユニットにおいて、共同して（関連して）処理が実行されることがあるので、関連する処理があるか否かを、他のユニットが識別できる必要があるからである。

#### 【 0 1 0 9 】

バス 2 に接続されている、ユニット間において、識別可能なものであれば、例えば、オブジェクト I D をイベントが発生する毎に、番号 1 から順番に順次設定するようにすることも可能である。しかしながらそのようにすると、所定のユニットが、所定のイベントに関して、オブジェクト I D を設定しようとした場合、バス 2 に接続されている全てのユニットが、既に設定しているオブジェクト I D を読み出して、競合するものがあるか否かを判定し、競合しないものに設定する必要が生じる。しかしながらこの例においては、G U I D がオブジェクト I D に含まれており、G U I D の値は、ユニット毎に異なるものであることが保証されている。したがって各ユニットは、レコード I D を自分自身の内部において、競

合するものがないように設定すれば、GUIDとレコードIDを組み合わせて得られるオブジェクトIDは、他のユニットが設定したオブジェクトIDと競合することはない。そこで各ユニットは、図32のフローチャートに示すような処理を行って、RSBに登録するイベントに対するオブジェクトIDを設定する。

【0110】

ステップS41において、コントローラ11は、オブジェクトID(=Global Unique ID+record ID)のうちのレコードIDとして、IRD1内において、その予約情報を一意に識別させるための仮IDを発生する。

【0111】

ステップS42において、コントローラ11は、IRD1のBBS14内のRSBに登録されているイベントの中から、1つのイベントを抽出して、そのイベントに対応するオブジェクトIDの中のレコードIDを抽出する。ステップS43において、コントローラ11は、ステップS41で発生した仮IDと、ステップS42で抽出したレコードIDが一致するか否かを判定し、一致する場合、ステップS41に戻り、仮IDを変更し、一致しない仮IDが発生されるまで、ステップS41乃至S43の処理を繰り返す。

【0112】

仮IDとステップS42で抽出したレコードIDが一致しないと判定された場合、ステップS44において、コントローラ11は、関連するサブユニットに関するRSBに公開されている全てのイベントを抽出したか否かを判定し、まだ抽出していないイベントが残っている場合はステップS42に戻り、全てのイベントを読み出したと判定するまで、ステップS42乃至S44の処理を繰り返す。全てのイベントを読み出したと判定された場合、ステップS45において、コントローラ11は、IRD1の機器ID(GUID)に、ステップS41で発生した仮ID(レコードID)を付加して、オブジェクトIDを生成し、RSBに記述する。

【0113】

このように、オブジェクトIDは、関連するサブユニットに関するRSBをチェックするだけで(関連しないサブユニットを有するユニットのRSBをチェッ

クすることなく) 設定することができる。これは、上述したように、他のユニットのRSBにおけるオブジェクトIDは、そのユニットのGUIDが含まれているため、他のユニットが設定したオブジェクトIDは、自分自身が設定したオブジェクトIDと一致することが有り得ないからである。したがって、オブジェクトIDを簡単に設定することが可能となる。

#### 【0114】

本実施の形態において、上述した一連の処理は、ハードウェアにより実行させることもできるが、ソフトウェアにより実行させることもできる。一連の処理をソフトウェアにより実行させる場合には、そのソフトウェアを構成するプログラムが、専用のハードウェアとしてのコントローラに組み込まれているか若しくは当該プログラムがインストールされた、例えば汎用のパーソナルコンピュータなどにより各種の機能を実行することが可能となる。

#### 【0115】

汎用のパーソナルコンピュータ101は、例えば、図33に示すように、CPU (Central Processing Unit) 111を内蔵している。CPU111には、バス115を介して入出インターフェース116が接続されており、CPU111は、入出インターフェース116を介して、ユーザから、キーボード、マウスなどよりなる入力部118から指令が入力されると、それに対応して、ROM (Read Only Memory) 112あるいはハードディスク114などの記録媒体、または、ドライブ120に装着された磁気ディスク131、光ディスク132、光磁気ディスク133などの記録媒体から、それらに記録されている、上述した一連の処理を実行するプログラムを読み出し、RAM (Random Access Memory) 113に書き込み、実行する。なお、ハードディスク114に格納されているプログラムには、予め格納されてユーザに配布されるものだけでなく、衛星もしくは、ネットワークから転送され、通信部119により受信されてダウンロードされたプログラムも含まれる。

#### 【0116】

また、CPU111は、プログラムの処理結果のうち、画像信号を、入出インターフェース116を介して、LCD (Liquid Crystal Display), CRT

(Cathode Ray Tube) などよりなる表示部 1 1 7 に出力する。

【 0 1 1 7 】

なお、本発明は、上述の説明した実施の形態に限定されるものではなく、本発明の精神を逸脱することなく種々の変形が可能とされるものである。例えば本実施の形態では、I R D 1 のコントローラ 1 1 が、D V C R 3 の B B S 3 4 の R S B 6 1 への前記テキスト情報の書き込みと読み出しを制御しているが、さらに他の機器（ユニット）のコントローラが当該 D V C R 3 の B B S 3 4 の R S B 6 1 へのテキスト情報の書き込みと読み出しを制御することも可能である。このとき、本実施の形態によれば、例えば D C V R 等で確認画面を出すような場合に、予約の内容を R S B の information block の形式としているので、他の機器による予約の概要を、上記 D V C R の確認画面上で、本体装置や外部装置の区別無く、同じ情報量でユーザに表示することができる。

【 0 1 1 8 】

【発明の効果】

本発明の情報処理装置及び方法、媒体においては、機能実行手段の使用予定に関する第 1 の情報に含まれない第 2 の情報を入手し、第 2 の情報を所定のブロック形式とし、第 1 の情報の付加情報として格納手段に格納させる制御を行うことにより、ユーザや各機器が予約情報等の詳細を入力でき、また、各機器が他の機器による予約情報等の詳細を入手でき、及び、各機器が他の機器に対して自己が持つ予約情報等の詳細を知らせることを実現可能となり、その結果、ダブルブッキングの発生を抑止可能となるだけでなく、分かり易く且つ使い易いネットワークシステムを構築可能である。

【 0 1 1 9 】

すなわち、本発明によれば、例えば予約のダブルブッキングが起きるときの警告表示を出すときに、そのダブルブッキングの発生原因をより分かり易い形でユーザに知らせることができる。また、本発明によれば、例えば仮の予約を可能とし、仮予約なので上書きしても良いという情報をエントリ時に書き込むことにより、後から予約するユーザがより適切な行動、つまり仮予約されていたものを消して新たな予約を行ったりする等の行動を取ることができるようになる。さらに

、例えばDCVR等で確認画面を出す場合は、その予約の概要を表示することが一般に行われるが、本発明によれば、予約の内容（第2の情報）を所定のブロック形式とすることで、他の機器による予約の概要を、確認画面上で本体装置、外部装置の区別無く、同じ情報量でユーザに表示することができる。

【図面の簡単な説明】

【図1】

本発明実施の形態のネットワークシステムの構成例を示すブロック図である。

【図2】

ターゲットデバイスのBBSのRSBをポスティングデバイスが制御する際の説明に用いる図である。

【図3】

BBSのRSBのフォーマットを説明する図である。

【図4】

図3のWrite Enabled list\_specific\_informationのフォーマットを説明する図である。

【図5】

図4のboard\_typeのフォーマットを説明する図である。

【図6】

図3のobject\_entryのフォーマットを説明する図である。

【図7】

図6のResource Schedule Entryのフォーマットを説明する図である。

【図8】

図7のstart\_timeのフォーマットを説明する図である。

【図9】

図7のDurationのフォーマットを説明する図である。

【図10】

図7のrepeat\_typeのフォーマットを説明する図である。

【図11】

スケジュールが週毎に繰り返される場合のrepeat\_informationのフォーマット



を説明する図である。

【図 1 2】

スケジュールが所定の間隔で行われる場合のrepeat\_informationのフォーマットを説明する図である。

【図 1 3】

図 7 のInfo blocksのフォーマットを説明する図である。

【図 1 4】

図 7 に示すResource Schedule Entryのcharacter\_code\_information\_blockのフォーマットを説明する図である。

【図 1 5】

図 7 に示すResource Schedule Entryのlanguage\_code\_information\_blockのフォーマットを説明する図である。

【図 1 6】

図 7 に示すResource Schedule Entryのraw\_text\_information\_blockのフォーマットを説明する図である。

【図 1 7】

図 1 6 に示すraw\_text\_information\_block内に記述されるraw\_text\_dataの一例を説明する図である。

【図 1 8】

図 1 のネットワークシステムの動作のうち、使用したい機器のRSBが書き込み可能かどうか調べる手順を説明するフローチャートである。

【図 1 9】

図 1 のネットワークシステムの動作のうち、使用予定のリソースが他の機器から使われる予定になっているかどうか確認する手順を説明するフローチャートである。

【図 2 0】

図 1 のネットワークシステムの動作のうち、使用予定を書き込む時の手順を説明するフローチャートである。

【図 2 1】

ライトオープンコマンドのフォーマットを説明する図である。

【図 2 2】

リードコマンドのフォーマットを説明する図である。

【図 2 3】

クリエイトコマンドのフォーマットを説明する図である。

【図 2 4】

図 1 2 の subfunction\_1 を説明する図である。

【図 2 5】

図 2 4 の subfunction\_1 の詳細を説明する図である。

【図 2 6】

図 2 5 におけるフィールドの値の例を示す図である。

【図 2 7】

ライトディスクリプタコマンドのフォーマットを説明する図である。

【図 2 8】

B B S のフォーマットを説明する図である。

【図 2 9】

B B S の root\_object\_list\_ID を説明する図である。

【図 3 0】

図 2 8 の B B S の supported\_board\_type\_specific\_information のフォーマットを説明する図である。

【図 3 1】

クローズコマンドのフォーマットを説明する図である。

【図 3 2】

オブジェクト I D 設定処理を説明するフローチャートである。

【図 3 3】

コンピュータの構成例を示すブロック図である。

【図 3 4】

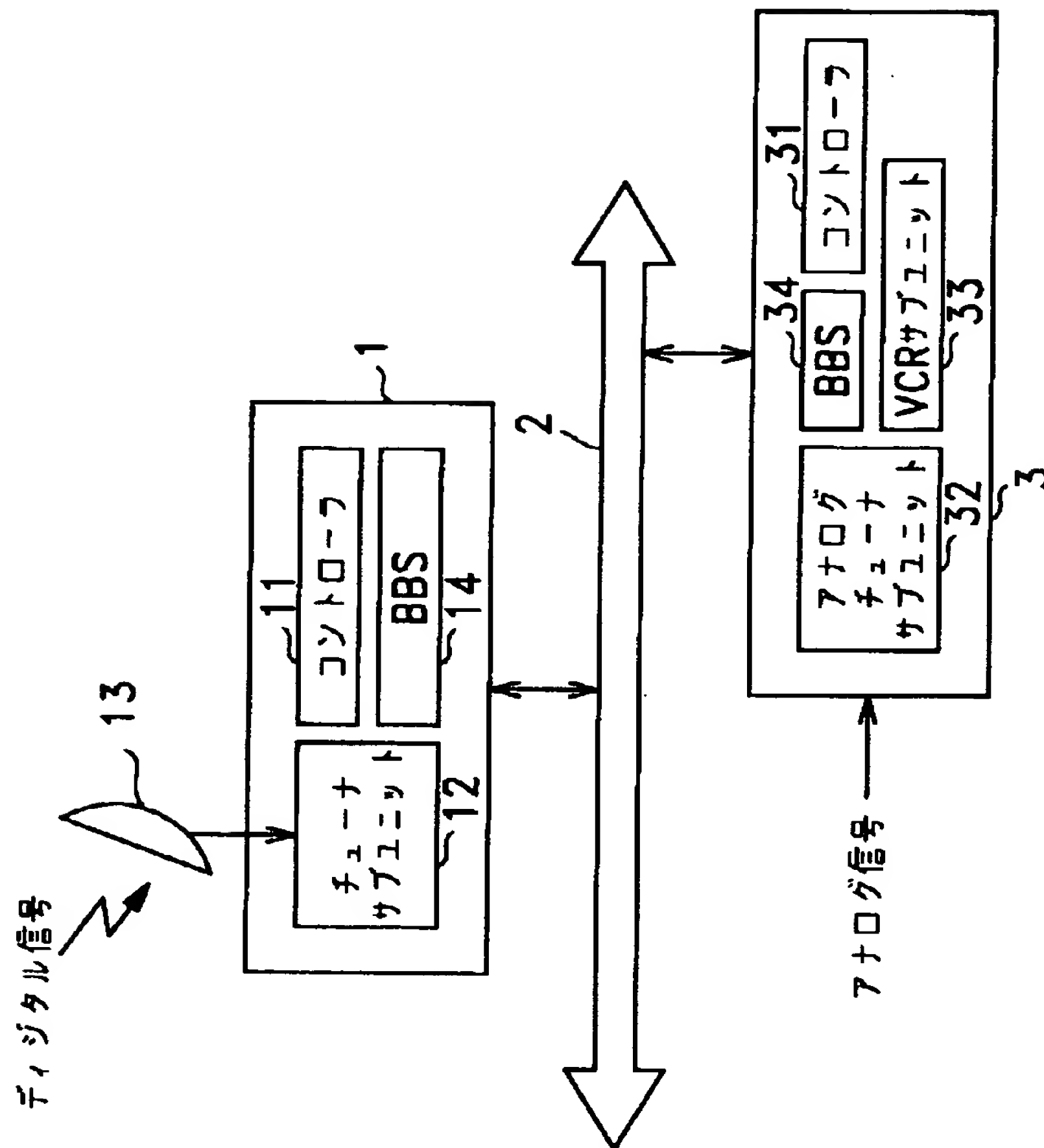
従来のネットワークシステムの構成例を示すブロック図である。

【符号の説明】

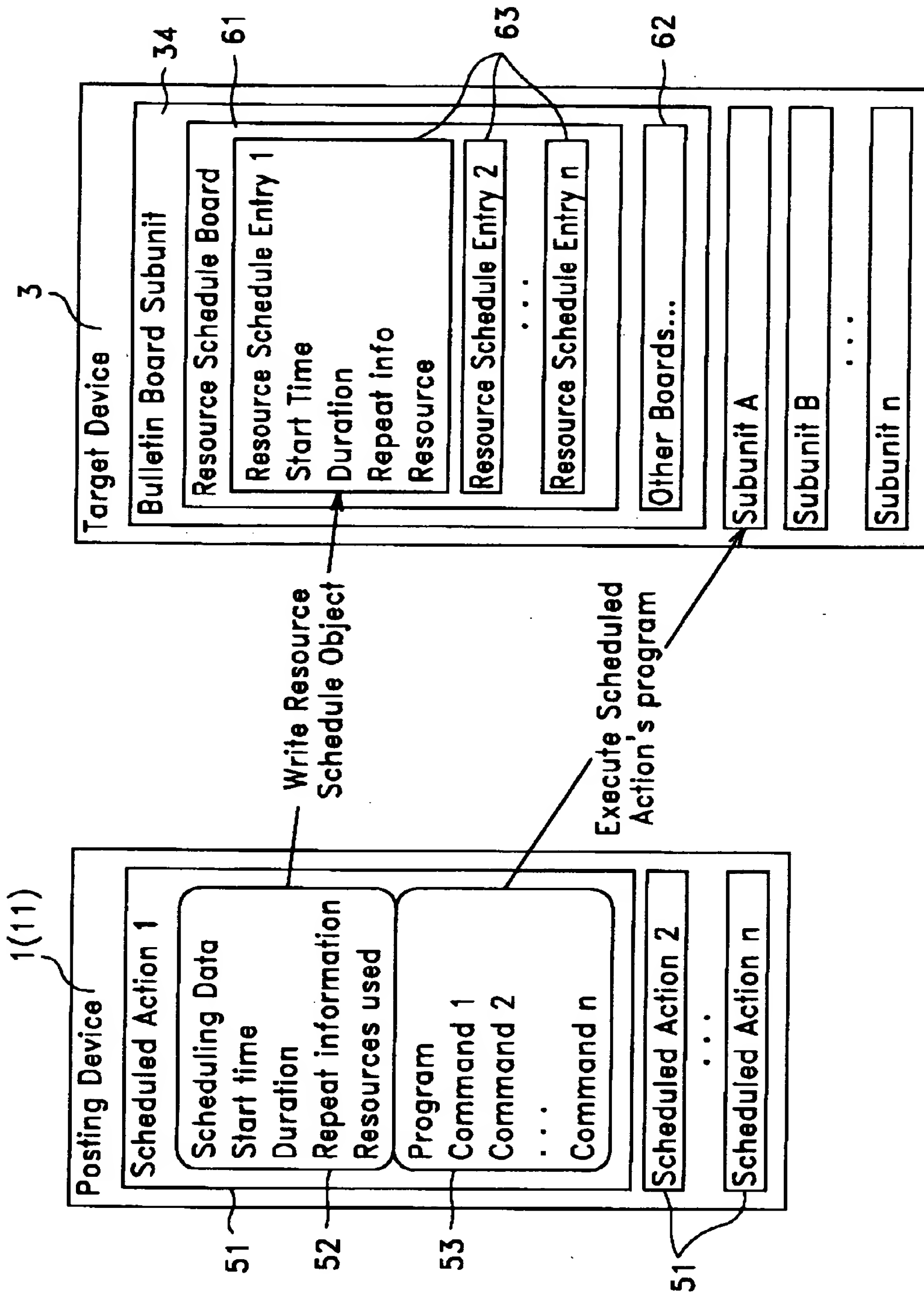
1    I R D、    2    I E E E 1 3 9 4 シリアルデータバス、    3    D V C R、  
1 1    コントローラ、    1 2    チューナサブユニット、    1 4    B B S、    3  
1    コントローラ、    3 2    アナログチューナサブユニット、    3 3    V C R サ  
ブユニット、    3 4    B B S、    6 1    R S B

【書類名】 図面

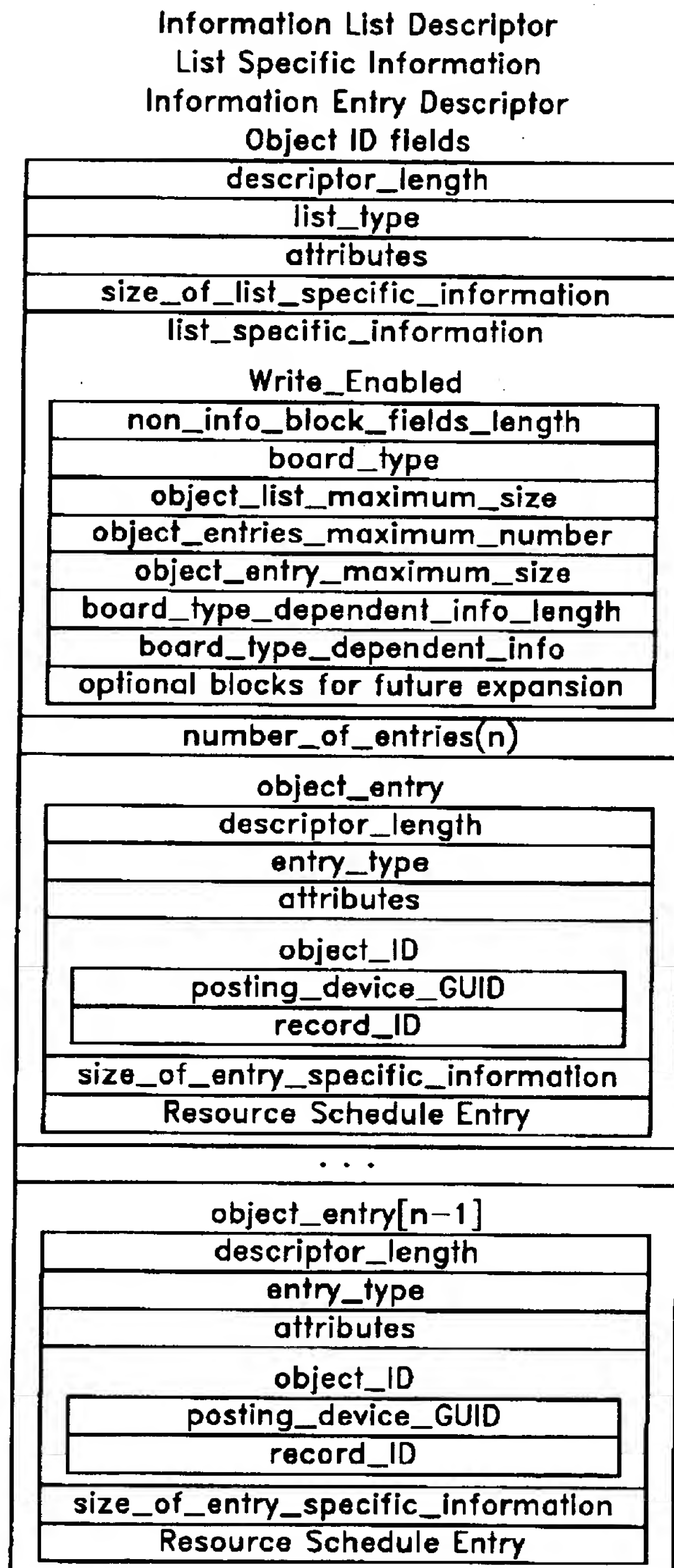
【図 1】



【図 2】



【図 3】





【図 4】

Address_offset	Contents
00 <sub>16</sub>	non_info_block_fields_length
01 <sub>16</sub>	
02 <sub>16</sub>	board_type
03 <sub>16</sub>	object_list_maximum_size
04 <sub>16</sub>	
05 <sub>16</sub>	object_entries_maximum_number
06 <sub>16</sub>	
07 <sub>16</sub>	object_entry_maximum_size
08 <sub>16</sub>	
09 <sub>16</sub>	board_type_dependent_information_length
0A <sub>16</sub>	
0B <sub>16</sub>	board_type_dependent_information
0C <sub>16</sub>	
0D <sub>16</sub>	
:	optional info blocks for future expansion
:	
:	

Write enabled list\_specific\_information fields for the Information List Descriptor

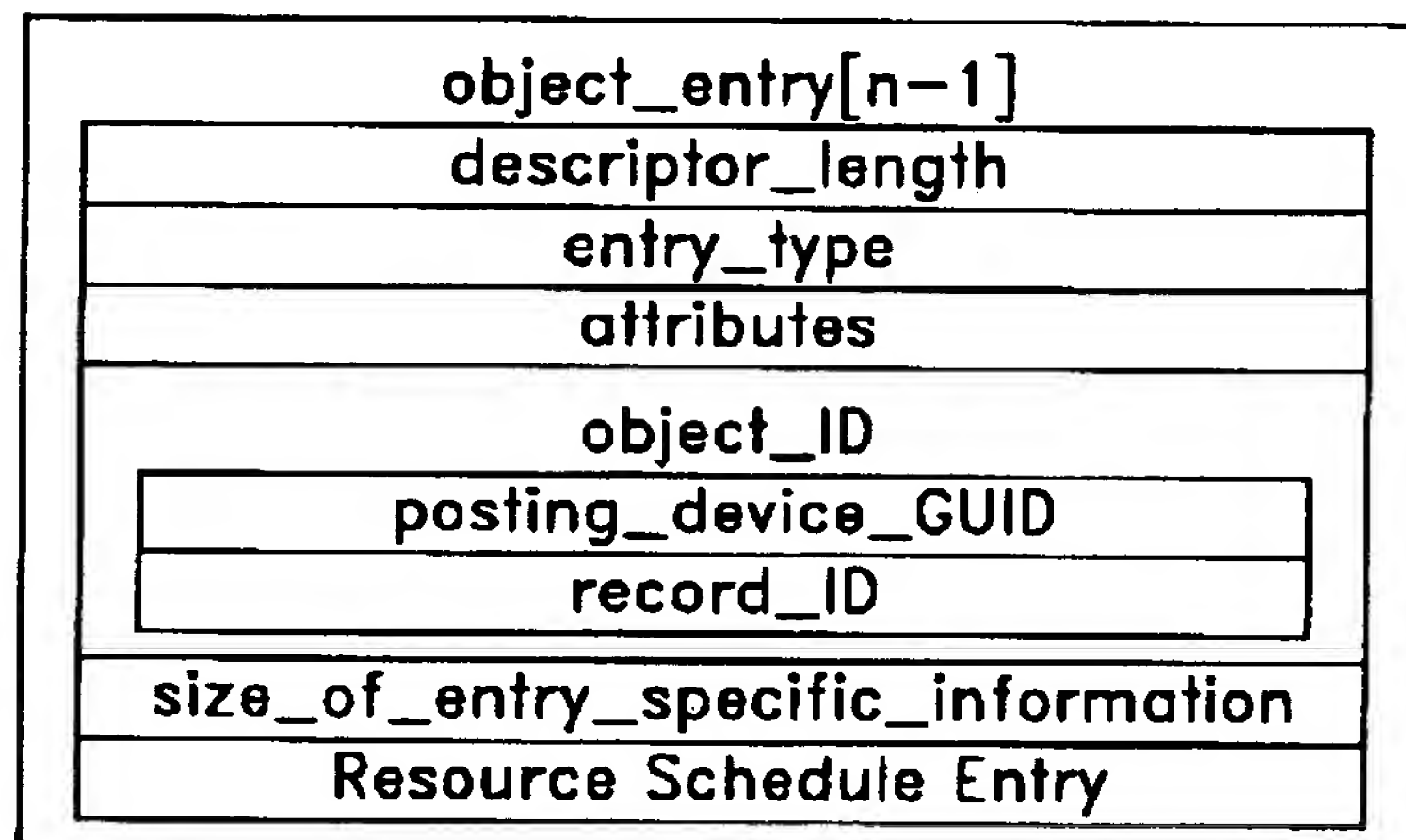
【図 5】

Existing board types

Value	Board type
00 <sub>16</sub>	Reserved
01 <sub>16</sub>	Resource Schedule Board
02 <sub>16</sub> - FF <sub>16</sub>	Reserved for future specification

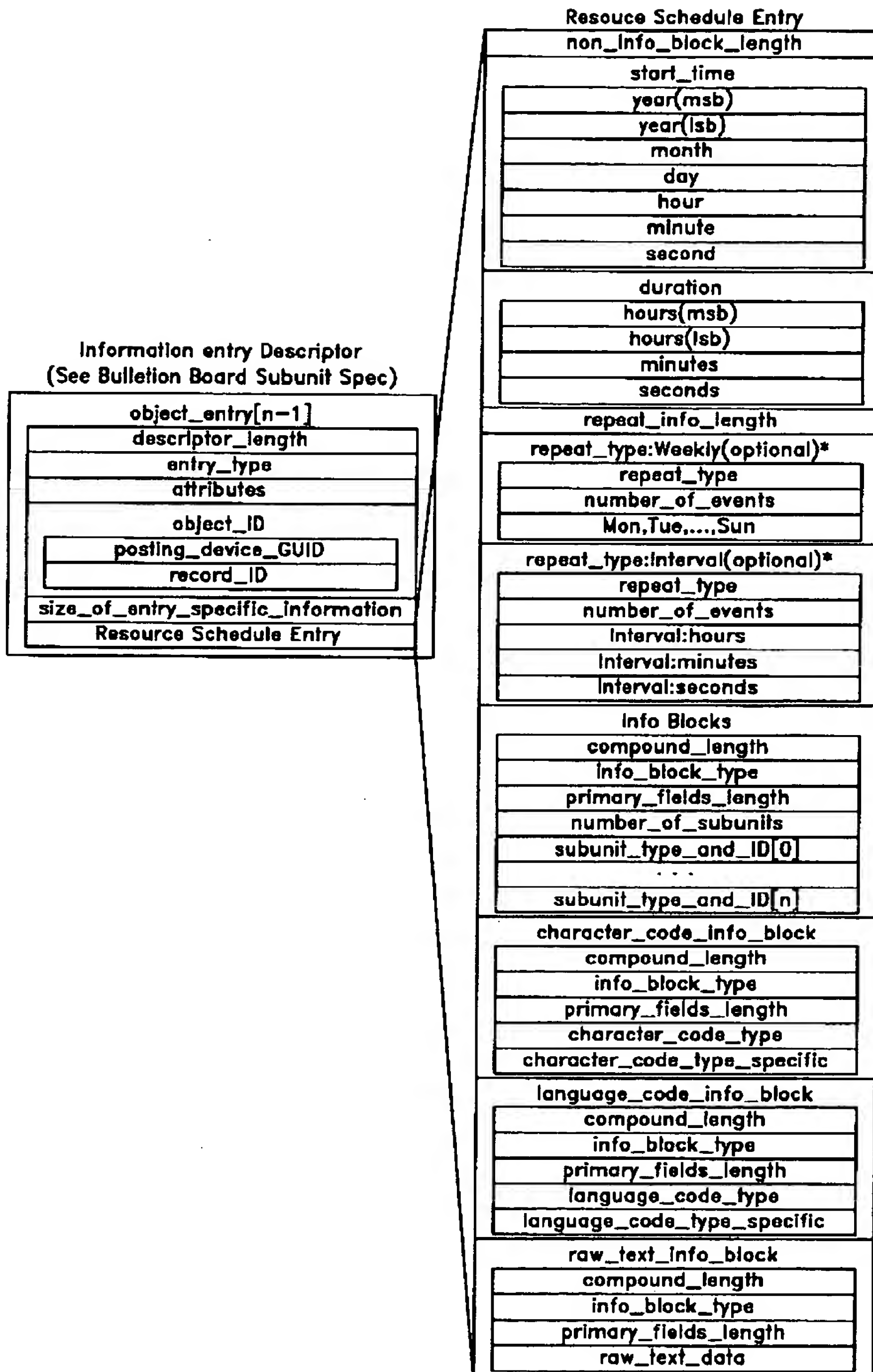
【図 6】

Information Entry Descriptor



Resource Schedule Entry high level view

【図 7】



【図 8】

Address_offset	Contents
00 <sub>16</sub>	year(msb)
01 <sub>16</sub>	year(lsb)
02 <sub>16</sub>	month
03 <sub>16</sub>	day
04 <sub>16</sub>	hour
05 <sub>16</sub>	minute
06 <sub>16</sub>	second

start\_time fields for Resouce Schedule Entries

【図 9】

Address_offset	Contents
00 <sub>16</sub>	Reserved(4 bits)   hours(msb)
01 <sub>16</sub>	hours(lsb)
02 <sub>16</sub>	minutes
03 <sub>16</sub>	seconds

duration fields for Resouce Schedule Entries

【図 1 0】

repeat\_type value assignment

Values	definition
00 <sub>16</sub>	Weekly schedule
01 <sub>16</sub> - 0F <sub>16</sub>	reserved
10 <sub>16</sub>	Interval schedule
0F <sub>16</sub> - FF <sub>16</sub>	reserved

【図 1 1】

	msb							lsb
address_offset	contents							
0E <sub>16</sub>	repeat_type							
0F <sub>16</sub>	number_of_events							
10 <sub>16</sub>	Sunday	Monday	Tuesday	Wed- nesday	Thurs- day	Friday	Saturday	Re- served

repeat\_information fields for Weekly

【図 1 2】

address_offset	contents	
0E <sub>16</sub>	repeat_type	
0F <sub>16</sub>	number_of_events	
10 <sub>16</sub>	Reserved(4 bits)	Interval:hours(msb)
11 <sub>16</sub>	interval:hours(lsb)	
12 <sub>16</sub>	interval:minutes	
13 <sub>16</sub>	interval:seconds	

repeat\_information fields for interval

【図 1 3】

address_offset	contents
00 <sub>16</sub>	compound_length
01 <sub>16</sub>	
02 <sub>16</sub>	info_block_type
03 <sub>16</sub>	
04 <sub>16</sub>	primary_fields_length
05 <sub>16</sub>	
06 <sub>16</sub>	number_of_subunits
07 <sub>16</sub>	subunit_type_and_ID[0]
:	:

Subunit\_Resource\_info\_block



【図 1 4】

character_code_info_block	
Address_offset	Contents
00 00 <sub>16</sub>	compound_length
00 01 <sub>16</sub>	
00 02 <sub>16</sub>	info_block_type=00 08 <sub>16</sub> (character_code_info_block)
00 03 <sub>16</sub>	
00 04 <sub>16</sub>	primary_fields_length
00 05 <sub>16</sub>	
00 06 <sub>16</sub>	character_code_type
00 07 <sub>16</sub>	character_code_type_specific
:	
:	

【図 1 5】

language_code_info_block	
Address_offset	Contents
00 00 <sub>16</sub>	compound_length
00 01 <sub>16</sub>	
00 02 <sub>16</sub>	info_block_type=00 09 <sub>16</sub> (language_code_info_block)
00 03 <sub>16</sub>	
00 04 <sub>16</sub>	primary_fields_length
00 05 <sub>16</sub>	
00 06 <sub>16</sub>	language_code_type
00 07 <sub>16</sub>	language_code_type_specific
:	
:	

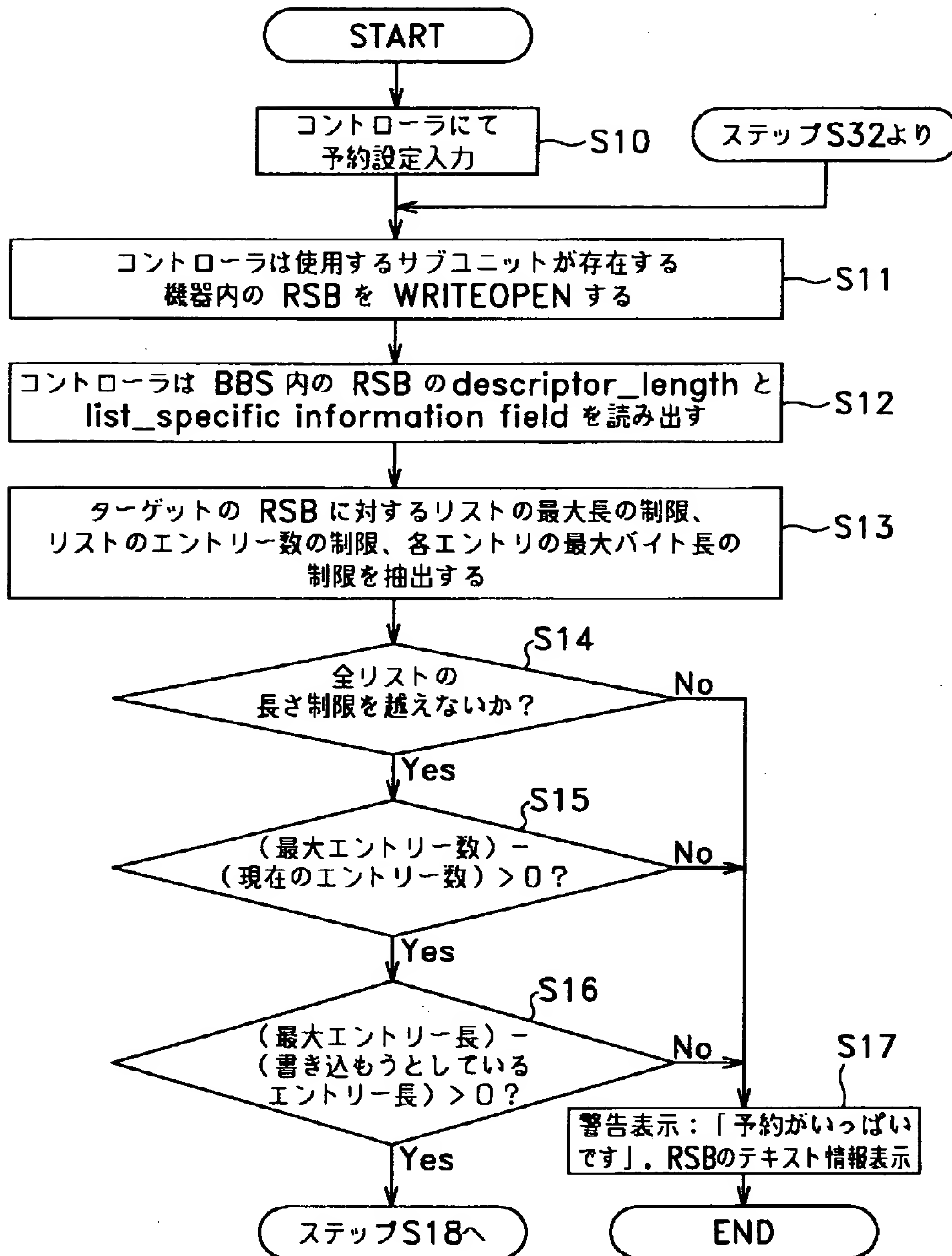
【図 1 6】

raw_text_info_block	
Address_offset	Contents
00 00 <sub>16</sub>	compound_length
00 01 <sub>16</sub>	
00 02 <sub>16</sub>	info_block_type=00 0A <sub>16</sub> (raw_text_info_block)
00 03 <sub>16</sub>	
00 04 <sub>16</sub>	primary_fields_length
00 05 <sub>16</sub>	
00 06 <sub>16</sub>	raw_text_data
:	
:	

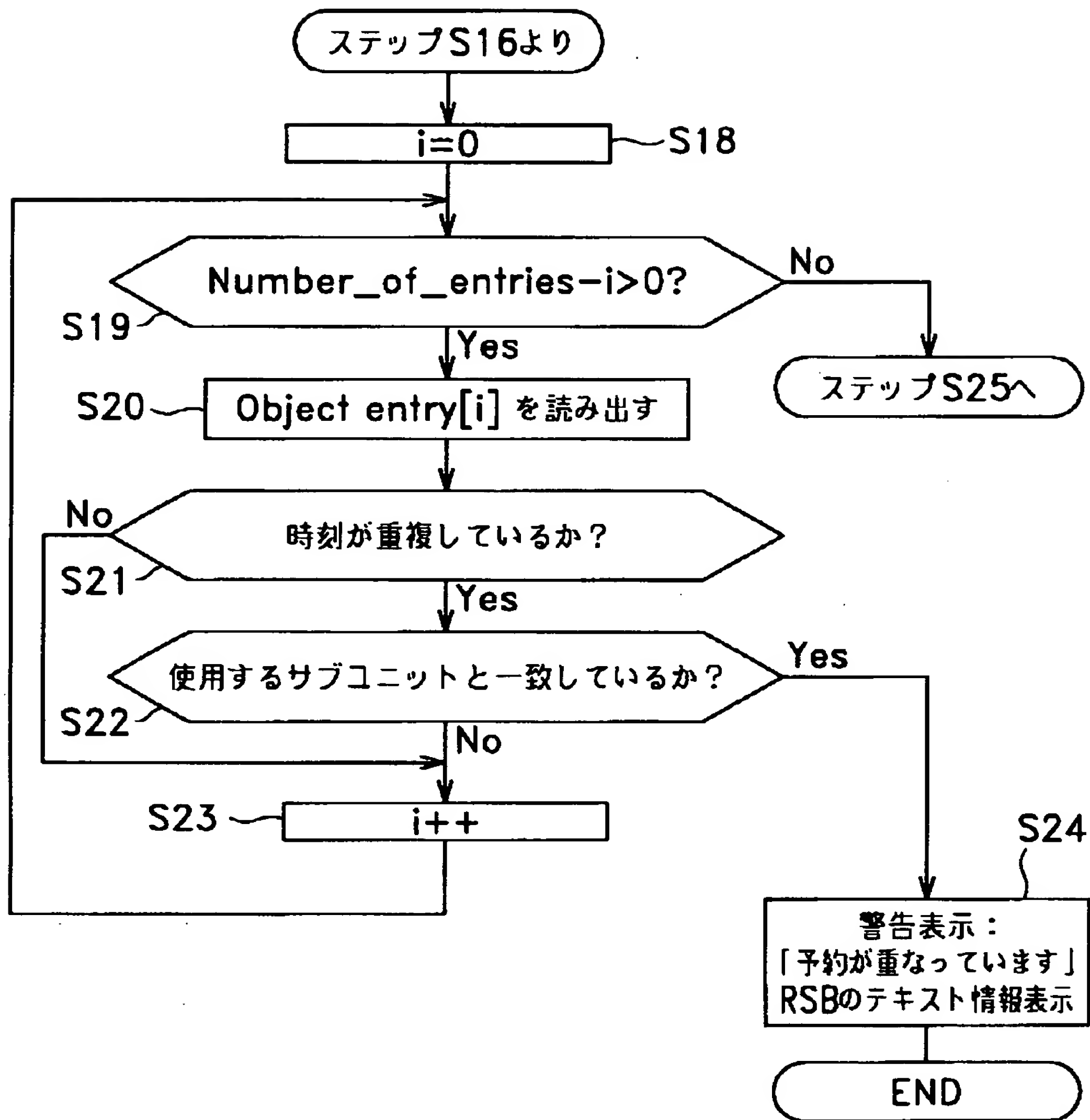
【図 1 7】

raw_text_data
チャンネル
プログラム名（番組）
制御情報（再生、録画、停止等）
備考（ペイパービュー）
プロバイダ名
仮予約中

【図 18】

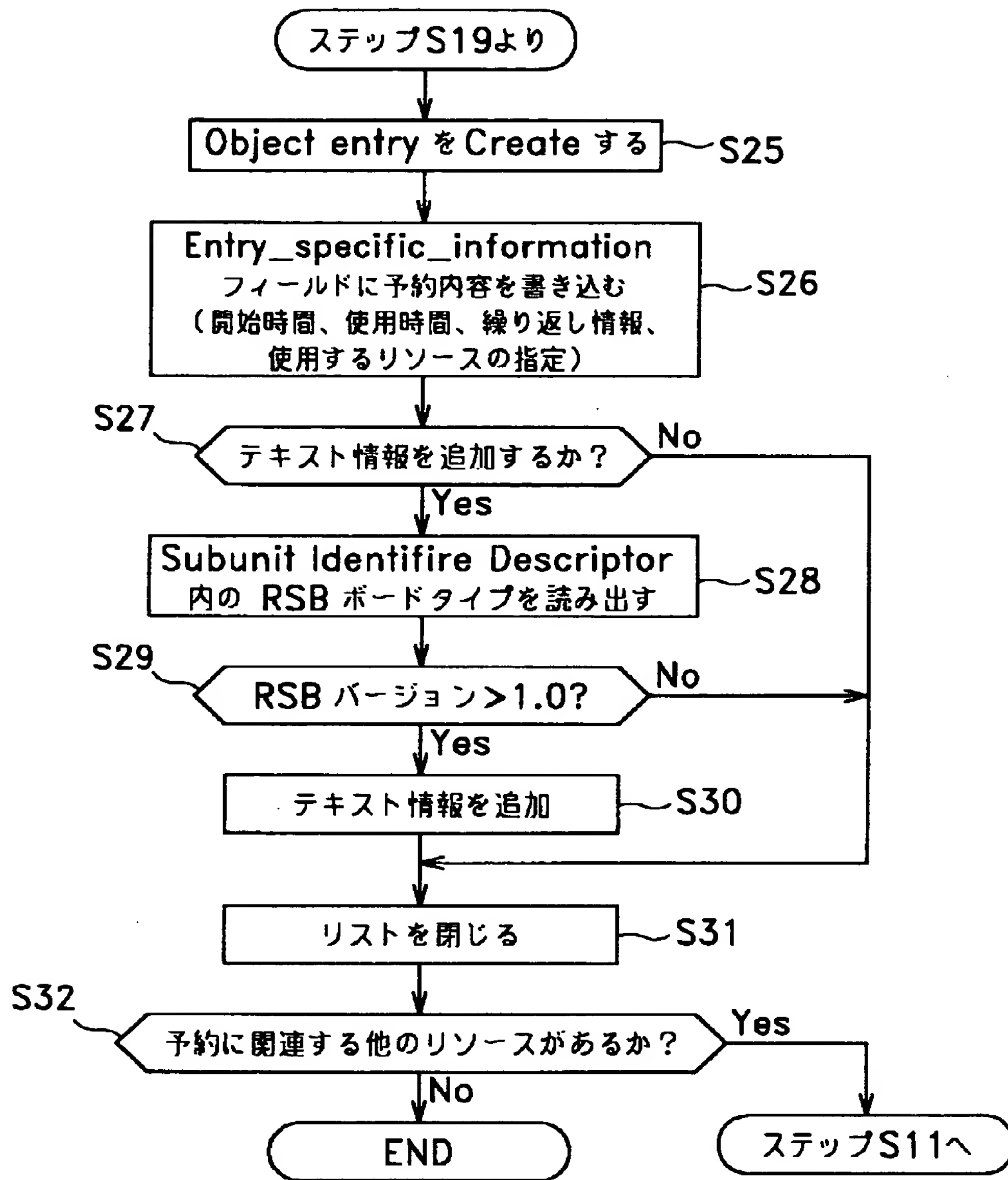


【図 19】





【図 2 0】



【図 2 1】

WRITE OPEN コマンド

opcode	OPEN DESCRIPTOR
operand 0	descriptor_type=10 <sub>16</sub>
operand 1	List ID:00 <sub>16</sub>
operand 2	List ID:01 <sub>16</sub>
operand 3	subfunction WRITE OPEN 03 <sub>16</sub>
operand 4	reserved 00 <sub>16</sub>

【図 2 2】

READ コマンド

	msb						lsb
opcode	READ DESCRIPTOR (09 <sub>16</sub> )						
operand 0	descriptor identifier						
operand 1	:						
:	:						
:	read_result_status						
:	reserved : 00 <sub>16</sub>						
:	data_length						
:	address						

【図 2 3】

	msb						lsb
opcode	CREATE DESCRIPTOR (00 <sub>16</sub> )						
operand 0	result						
operand 1	subfunction_1						
operand 2	reserved						
operand 3	subfunction_1_specification						
:							
:							

【図 2 4】

subfunction_1の値	meaning
00 <sub>16</sub>	create a new descriptor
01 <sub>16</sub>	create a new object and its child list
all other values	reserved for future specification

【図 2 5】

subfunction_1_specification for subfunction_1=01 <sub>16</sub>								
	msb							lsb
operand 3	descriptor_identifier_where							
:								
:								
:	descriptor_identifier_what_1							
:								
:								
:	descriptor_identifier_what_2							
:								
:								

【図 2 6】

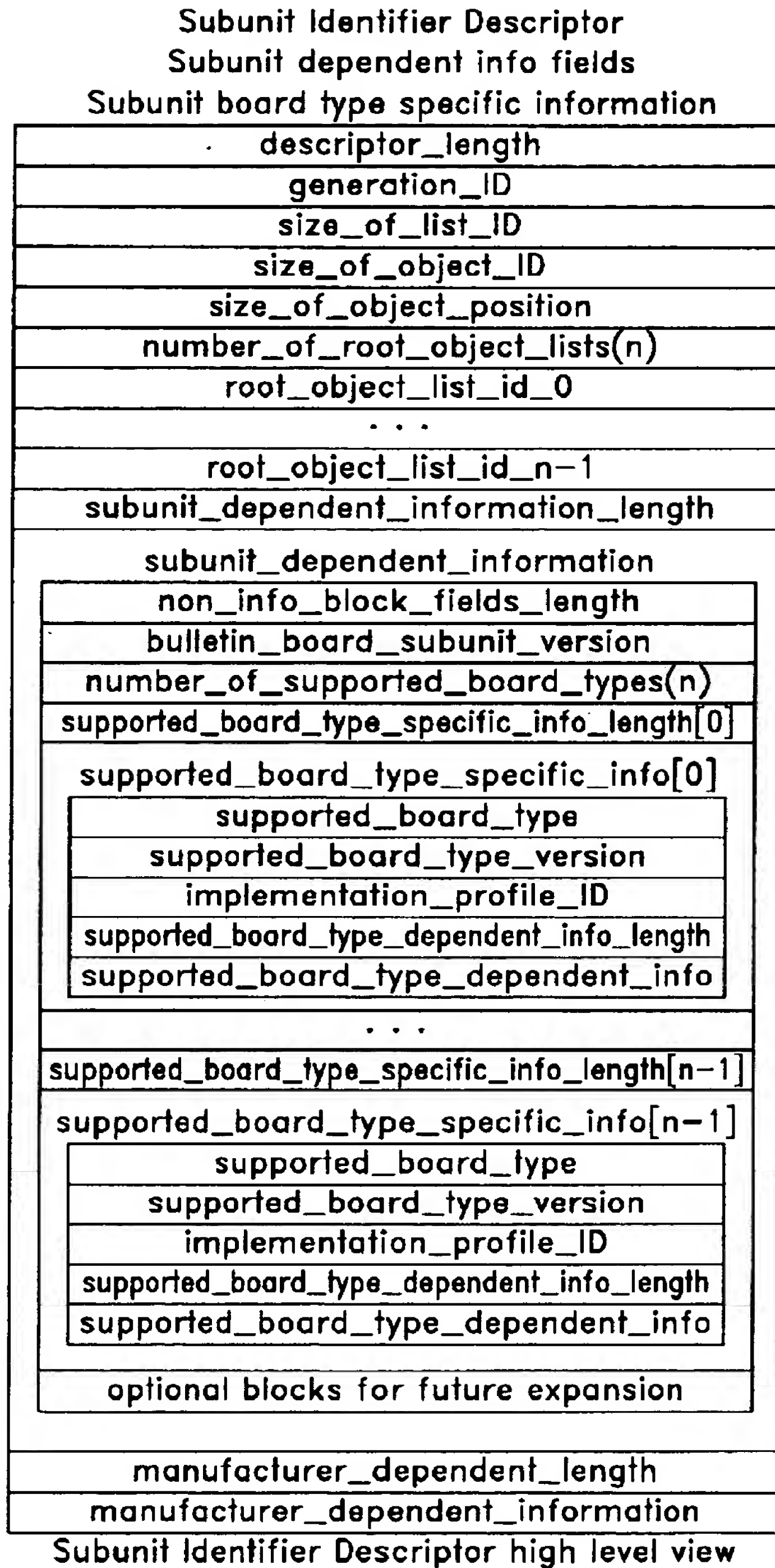
descriptor_type of descriptor_identifier_ where	descriptor_type of descriptor_identifier_ what_1	descriptor_type of descriptor_identifier_ what_2	meaning
20 <sub>16</sub>	22 <sub>16</sub>	11 <sub>16</sub>	Create an object and its child list.  create the new object and place it in the location specified by where,the entry_type is specified by what_1.Also create the new list as the child of the new object The list_type is specified by what_2.
all other values			reserved for future specification

【図 2 7】

WRITE DISCCRIPTOR コマンド

opcode	WRITE DESCRIPTOR (0A <sub>16</sub> )
operand 0	descriptor identifier
:	subfunction:partial_replace(50 <sub>16</sub> )
:	group_tag:immediate(00 <sub>16</sub> )
:	replacement_data_length
:	address
:	original_data_length
:	replacement_data

【図 2 8】



【図 2 9】

root\_object\_list\_ID Value Assignment

Value	List definition
1001 <sub>16</sub>	Resource Schedule List
1002-10FF <sub>16</sub>	reserved



【図 3 0】

Address_offset	Contents
00 <sub>16</sub>	supported_board_type
01 <sub>16</sub>	supported_board_type_version
02 <sub>16</sub>	implementation_profile_ID
03 <sub>16</sub>	supported_board_type_dependent_information_length
04 <sub>16</sub>	
05 <sub>16</sub>	supported_board_type_dependent_information
:	
:	

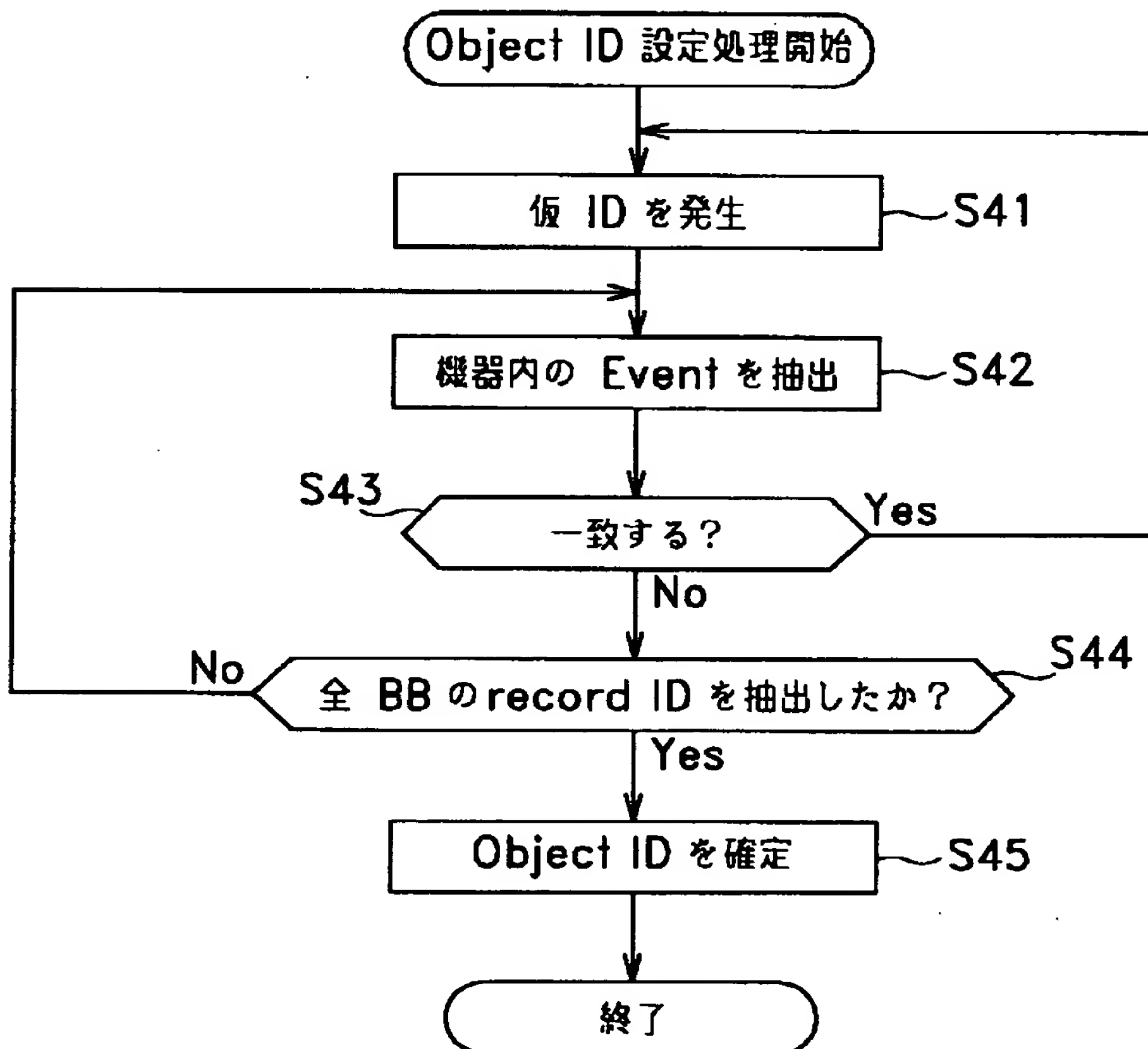
supported\_board\_type\_specific\_information fields

【図 3 1】

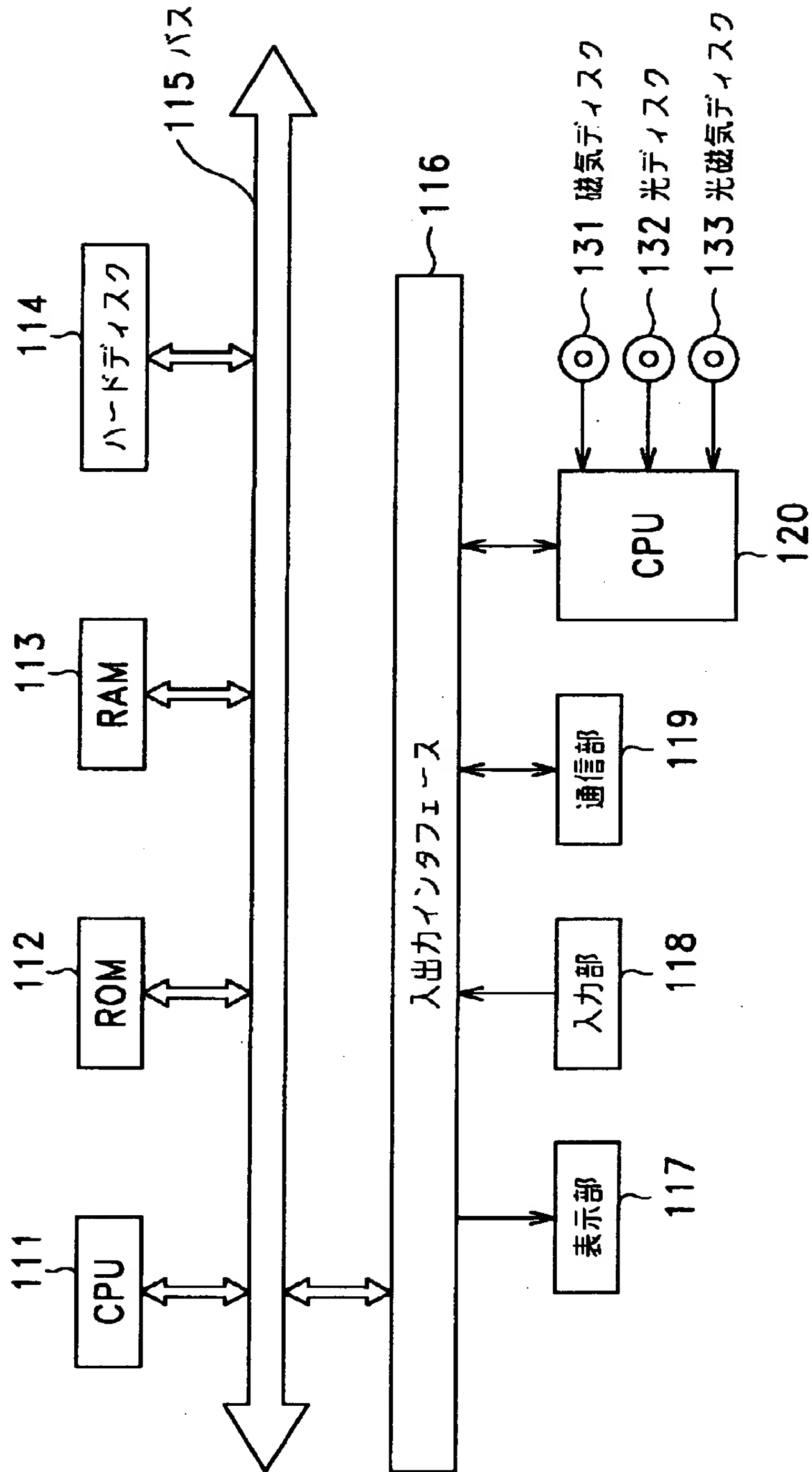
CLOSE コマンド

opcode	OPEN DESCRIPTOR
operand 0	descriptor type=10 <sub>16</sub>
operand 1	List ID:00 <sub>16</sub>
operand 2	List ID:01 <sub>16</sub>
operand 3	subfunction CLOSE 00 <sub>16</sub>
operand 4	reserved 00 <sub>16</sub>

【図 3 2】

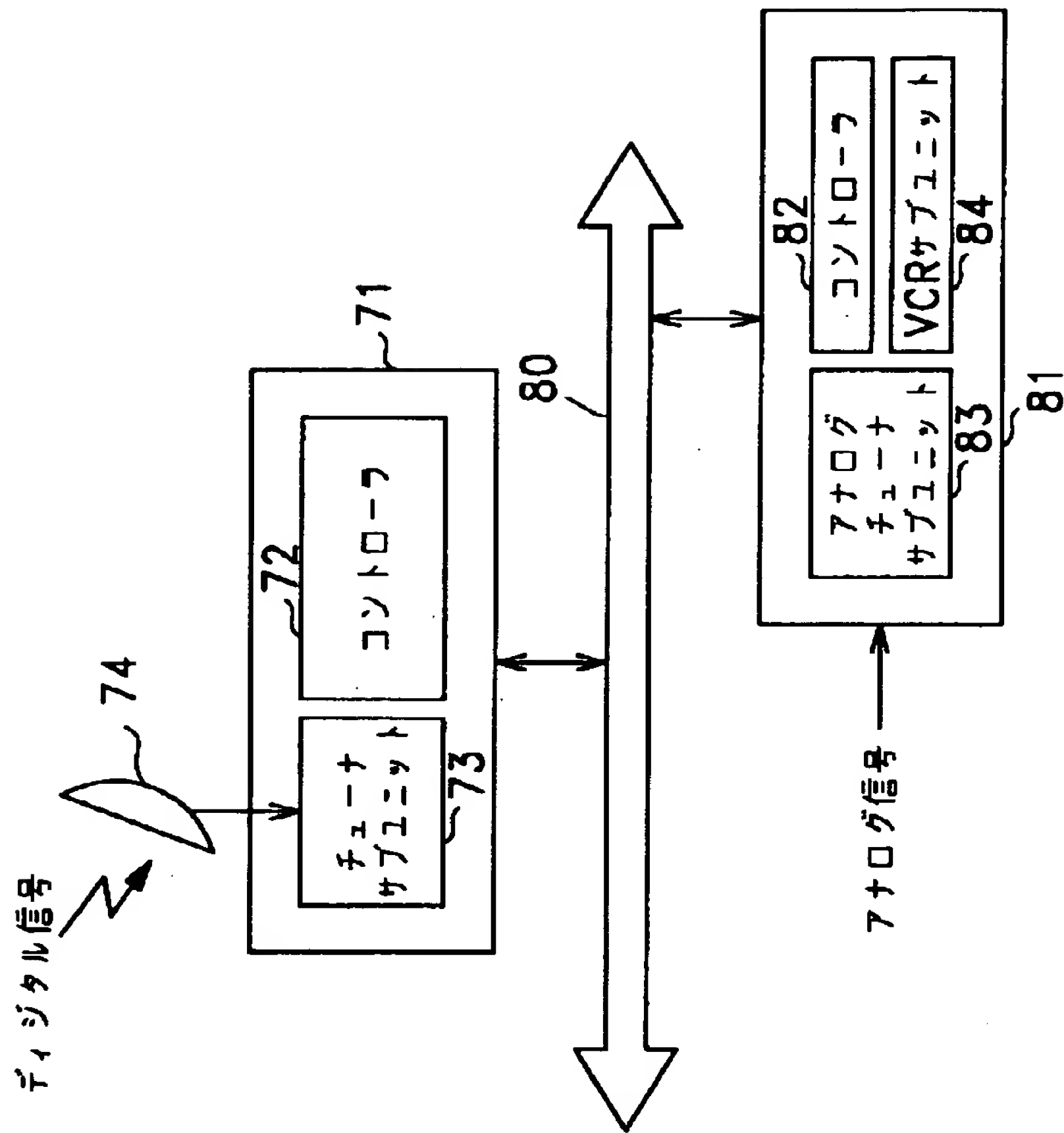


【図 33】



パーソナルコンピュータ 101

【図 34】



【書類名】            要約書

【要約】

【課題】    ユーザや各機器が予約情報等の詳細を入力でき、各機器が他の機器による予約情報等の詳細を入手でき、各機器が他の機器に対して自己が持つ予約情報等の詳細を知らせる手段を実現し、ダブルブッキングの発生を抑止し、分かり易く且つ使い易いネットワークシステムを構築可能とする。

【解決手段】    R S B の Resource Schedule Entry に、予約の詳細を表すローテキストを含む raw\_text\_information\_block と、ローテキストのコーディングを表す character\_code\_information\_block と language\_code\_information\_block を記述する。

【選択図】        図 7

出 願 人 履 歴 情 報

識別番号 [ 0 0 0 0 0 2 1 8 5 ]

1. 変更年月日 1 9 9 0 年 8 月 3 0 日  
[変更理由] 新規登録  
住 所 東京都品川区北品川 6 丁目 7 番 3 5 号  
氏 名 ソニー株式会社